USER GUIDE

Nuke 6.2v2

NUKE

BY THE FOUNDRY

Visual Effects Software

The Foundry

Nuke™ User Guide. Copyright © 2011 The Foundry Visionmongers Ltd. All Rights Reserved. Use of this User Guide and the Nuke software is subject to an End User License Agreement (the "EULA"), the terms of which are incorporated herein by reference. This User Guide and the Nuke software may be used or copied only in accordance with the terms of the EULA. This User Guide, the Nuke software and all intellectual property rights relating thereto are and shall remain the sole property of The Foundry Visionmongers Ltd. ("The Foundry") and/or The Foundry's licensors.

The EULA can be read in the Nuke User Guide, Appendix E.

The Foundry assumes no responsibility or liability for any errors or inaccuracies that may appear in this User Guide and this User Guide is subject to change without notice. The content of this User Guide is furnished for informational use only.

Except as permitted by the EULA, no part of this User Guide may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, recording or otherwise, without the prior written permission of The Foundry. To the extent that the EULA authorizes the making of copies of this User Guide, such copies shall be reproduced with all copyright, trademark and other proprietary rights notices included herein. The EULA expressly prohibits any action that could adversely affect the property rights of The Foundry and/or The Foundry's licensors, including, but not limited to, the removal of the following (or any other copyright, trademark or other proprietary rights notice included herein):

Nuke™ compositing software © 2011 The Foundry Visionmongers Ltd. All Rights Reserved.

Nuke™ is a trademark of The Foundry Visionmongers Ltd.

Digital Domain ® is a registered trademark of Digital Domain, Inc.

Primatte™ keyer tool © 1997-2011 Photron USA, Inc. All Rights Reserved.

Primatte™ is a trademark of IMAGICA Corp.

Primatte™ patent is held by IMAGICA Corp.

In addition to those names set forth on this page, the names of other actual companies and products mentioned in this User Guide (including, but not limited to, those set forth below) may be the trademarks or service marks, or registered trademarks or service marks, of their respective owners in the United States and/or other countries. No association with any company or product is intended or inferred by the mention of its name in this User Guide.

ACADEMY AWARD ® is a registered service mark of the Academy of Motion Picture Arts and Sciences.

Linux ® is a registered trademark of Linus Torvalds.

Windows ® is the registered trademark of Microsoft Corporation.

Mac, Mac OS, Tiger, Shake, Final Cut Pro and QuickTime are trademarks of Apple, Inc., registered in the U.S. and other countries.

Adobe ® and Photoshop ® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Maya ® is a registered trademark of Autodesk, Inc., in the USA and other countries.

Houdini ® is a registered trademark of Side Effects Software, Inc.

Boujou is a trademark of 2d3 Ltd.

3D-Equalizer is a trademark of Science.D.Visions.

FrameCycler ® is a registered trademark of Iridas, Inc. OpenGL ® is a trademark or registered trademark of Silicon Graphics, Inc., in the United States and/or other countries worldwide.

RenderMan ® is a registered trademark of Pixar.

Cineon™ is a trademark of Eastman Kodak Company.

Stereoscopic images courtesy of Mr. Henry Chung, HKSC (http://www.stereoscopy.com/henry/). Images illustrating warping and morphing courtesy of Ron Brinkmann (http://www.digitalcompositing.com). Images from “The Day After Tomorrow” ©2004 courtesy of and copyright by 20th Century Fox. Images from "Stealth" courtesy of and copyright by Sony Pictures Inc. Images from "xXx" ©2002 courtesy of and copyright by Columbia Pictures Industries. All rights reserved by their respective owners in the United States and/or other countries.

Thank you to Diogo Girondi for providing icons for Nuke user interface.

The Foundry
6th Floor
The Communications Building
Leicester Square
London
WC2R 7LT
UK

Rev: January 14, 2011

Contents

PREFACE

About this Manual ... 22
Getting Help ... 22
    Viewing Online Help ... 22
    Contacting Customer Support ... 23

GETTING STARTED

About the Chapters ... 24

INSTALLATION AND LICENSING

System Requirements ... 25
    Windows and Linux ... 25
    Mac OS X ... 26
Licensing Nuke ... 26
    Obtaining Licenses ... 26
    Installing Licenses ... 28
    Further Reading ... 29
Installing Nuke ... 29
    Installation on Windows ... 29
    Installation on Linux ... 30
    Installation on Mac OS X ... 32
Launching Nuke ... 33
    Launching the Commercial Version ... 33
    Launching the Nuke Personal Learning Edition (PLE) ... 35
About the Personal Learning Edition ... 36
    PLE Versus the Commercial Version of Nuke ... 36

THE NUKE PLUG-IN INSTALLER

What Is the Nuke Plug-in Installer ... 38
Accessing the Nuke Plug-in Installer ... 38
Getting an Overview of the Plug-ins ... 38
Viewing Details of Plug-ins ... 38
Downloading Plug-ins ... 38

USING THE INTERFACE

Understanding the Workflow ... 39
The Nuke Window ... 40
    Panes and Panels ... 40
    Tabbed Panels ... 41
    Toolbar, Menu Bar, and Content Menus ... 42

Using the Toolbar 44

Using the Menu Bar 45

Working with Nodes 46
  Adding Nodes 46
  Selecting Nodes 47
  Replacing Nodes 49
  Renaming Nodes 49
  Editing Nodes 50
  Cloning Nodes 51
  Disabling and Deleting Nodes 52
  Connecting Nodes 52
  Indicators on Nodes 55
  Searching for Nodes 56
  Viewing Information on Nodes 56
  Customizing the Node Display 57

Navigating Inside the Node Graph 58
  Panning 58
  Zooming 59
  Fitting Selected Nodes in the Node Graph 60
  Fitting the Node Tree in the Node Graph 60

Properties Panels 60
  Managing the Properties Bin 60
  Controls That Appear on All Properties Panels 61
  Displaying Parameters 63
  Using Input Fields 64
  Using Sliders 65
  Separating Channels 65
  Using the Color Picker and Color Controls 65
  Using Color Sliders and Color Wheel 67

Animating Parameters 71
  Working with Animated Parameters 72
  Animated Parameters in the Dope Sheet and the Curve Editor 73

Working with Keyframes in the Dope Sheet 74
  Viewing Keyframes in the Dope Sheet 74
  Synchronising the Frame Range Between the Dope Sheet and the Curve Editor 75
  Selecting and Editing Keyframes. 75
  Repositioning and Trimming Read Nodes 76

Using the Curve Editor 77
  Displaying Curves 77
  Editing Curves 78

Viewers 87
  Adding Viewer Nodes 88

  Connecting Viewer Nodes 88
  Toggling Views 89
  Panning and Zooming the Viewer Window 89
  Hiding Floating Viewers 90
  Using the Viewer Controls 90
  Using the Viewer Composite Display Modes 108
  Hiding and Showing Viewer Toolbars 110
  Locking the Viewer Zoom Level 110
Using the File Browser 111
Undoing and Redoing 114
Progress Bars 115
Handling Errors 115
Customizing the Interface 116
  Interface Layouts 116

MANAGING SCRIPTS
Working with Multiple Image Formats 121
8-, 16-, and 32-Bit Image Processing 121
Setting Up Your Script 122
  Name, Timespan, and Frame Rate 122
  Setting the Default Project Directory 122
  Full-size Formats 122
  Proxy Mode 124
Image Caching 129
  The Cache Directory 129
  Defining the Settings for Caching 129
  Clearing the Disk Cache 130
  Using the DiskCache Node 130
Saving Scripts and Recovering Backups 132
  Saving Scripts 132
  Automatic Backup of Scripts 132
  Recovering Backups 134
Loading Image Sequences 134
  Loading Images from an External File Browser 136
  Naming Conventions 136
  Changing the Relation Between the Current Frame and the Frame Read In 136
Reformatting Image Sequences 139
Loading Nuke Scripts 139
Closing Nuke Scripts 140
Defining Frame Ranges 140
File Name Search and Replace 141
Displaying Script Information 142

Grouping Nodes in the Node Graph 142
  Grouping Nodes with the Backdrop Node. 142
  Grouping Nodes with the Group Node 145
Adding Notes to the Node Graph 146
Using the Precomp Node 147
  Creating Precomp Nodes 147
  Using a Precomp Node to Speed-up Rendering 149
  Precomp Revisions 151
  Collaborative Workflow Example 151
Working with File Metadata 152
  Metadata in Nuke 152
  Viewing Metadata. 153
  Comparing Metadata Between Inputs. 154
  Modifying Metadata 154
  Copying Metadata from One Input to Another and Filtering Metadata . 157
  Adding a Timecode to Metadata. 157
  Rendering Metadata 158
  Accessing Metadata via TCL Expressions. 159
  Accessing Metadata via Python 159

REFERENCE

Organisation of the Section 160

REFORMATTING ELEMENTS

Reformatting Images 162
  Using the Reformat Node 162
  Cropping Elements 166
Adjusting the Bounding Box 168
  Resizing the Bounding Box 168
  Copying a Bounding Box from One Input to Another. 169
  Adding a Black Outside Edge to the Bounding Box 170

CHANNELS

Overview 172
  Understanding Channels 172
  Understanding Channel Sets (Layers) 172
Creating Channels and Channel Sets 173
Calling Channels 174
  Selecting Input Channels. 175
  Selecting Masks 176
Linking Channels 178
Tracing Channels. 178
Renaming Channels. 179

Removing Channels and Channel Sets 179
Swapping Channels 180
    Channels from Input 1 180
    Channels from Input 2 180
    Channel Outputs 181
    Assigning Constants 181
    Creating Swap Channel Sets 182
    In Summary 182

MERGING IMAGES

Layering Images Together with the Merge Node 183
    Merge Operations 185
Generating Contact Sheets 196
Copying a Rectangle from one Image to Another. 198

COLOR CORRECTION AND COLOR SPACE

Making Tonal Adjustments 203
    Using Histograms 203
    Sampling White and Black Points 204
    Making Basic Corrections 205
    Using Sliders 206
    Using Color Curves 206
Making Hue, Saturation, and Value Adjustments 209
    Correcting HSV 211
    Correcting Hue 212
    Correcting Saturation 214
Masking Color Corrections 214
Applying Grain. 216
    Using Synthetic Grain 216
    Using Practical Grain. 217
Applying Mathematical Operations to Channels 220
    Clamping Channel Values 220
    Offsetting Channel Values 221
    Inverting Channel Values 221
    Multiplying Channel Values. 222
    Applying Expressions to Channel Values .222
Transforming the Color Space 223
    Overriding the Default Cineon Conversion 224
    Making Other Color Space Conversions 224
    Changing the Viewer Color Space. 225

TRANSFORMING ELEMENTS

Transforming in 2D. 226
    Using the 2D Transformation Overlay 226
    Choosing a Filtering Algorithm 227
    How Your Nodes Concatenate 231

- Translating Elements. 231
- Rotating Elements. 232
- Scaling Elements 233
- Skewing Elements 235

Applying Core Transformations in 2.5D 236
- Adding a Card3D Node 236
- Specifying the Order of Operations 236
- Choosing a Filtering Algorithm 237
- Using the 3D Transformation Handles 237
- Translating Elements. 238
- Rotating Elements. 238
- Scaling Elements 239
- Skewing Elements 239

Adding Motion Blur. 240

Replicating the Input Image Across the Output. 244

TRACKING AND STABILIZING

Tracking an Image. 246
- Activating Track Anchors 248
- Positioning Track Anchors 249
- Calculating the Track 250

Retracking Part of a Track 251

Editing Tracks 251
- Manipulating the Track Overlays 252
- Manipulating Track Curves and Smoothing Tracks. 252
- Tracking and Multiview Projects. 254

Applying Tracking Data. 255
- Applying Tracking Data Using Tracker Controls. 255
- Applying Tracking Data via Linking Expressions 256

KEYING WITH PRIMATTE

Accessing Primatte from Nuke 261

Primatte Basic Operation Tutorial 261
- Auto-Compute. 262
- Select BG Color 263
- Clean BG Noise 264
- Clean FG Noise 265
- Spill Removal - Method #1 266
- Spill Removal - Method #2 267
- Spill Removal - Method #3 268

Repeatable Sampling Tools. 268
- The Spill Sampling Tools 269
- The Matte Sampling Tools 269
- The Detail Sampling Tools. 270

Spill Replacement Options 270

No Suppression (No Suppression) 270
Complemental Replacement Mode (Complement) 271
Solid Color Replacement Mode (Solid Color). 271
Defocus Spill Replacement (Defocused Background) 272
Primatte Tools and Buttons 273
Initialize Section 273
Degrain Section. 278
Actions Section 281
Fine Tuning Section. 284
Spill Process Section. 286
Output Section 286
The Primatte Algorithm. 287
Explanation of How Primatte Works. 287
Explanation of How Primatte RT+ works 295
Explanation of How Primatte RT works 296
Contact Details 297
Main Office 297
Primatte Office 297
Proprietary Notices 297

KEYING WITH KEYLIGHT

Quick Key 298
Basic Keying 299
Picking the Screen Color 299
Screen Matte. 300
Viewing the Key 300
Keying More 301
Advanced Keying. 301
Under the Hood. 302
View 302
Screen Color 304
Clip Black and White 309
Screen Gain 310
Screen Balance 311
PreBlur 312
Tuning 312
Screen Processing. 312
Mattes. 315
Inside and Outside Masks. 315
Source Alpha 316
Color Replacement 317

KEYING WITH ULTIMATTE

Ultimatte Quick Start 319
Connecting the Ultimatte Node 319

Sampling the Screen Color 320
Using Overlay Tools and Screen Correct 321
Adjusting the Density of the Matte 323
Adjusting Spill Controls 324
Retaining Shadows and Removing Noise 325
Adjusting Color Controls. 326
Adjusting Film Controls. 327
Choosing an Output Mode 327

USING ROTOPAINT

Roto or RotoPaint?. 329
RotoPaint Quick Start. 329
Connecting the RotoPaint Node 330
Working with the Toolbars 331
Working with the Stroke/Shape List 331
Drawing Paint Strokes 333
    Using the Brush tool 335
    Using the Eraser Tool 335
    Using the Clone Tool. 336
    Using the Reveal Tool 337
    Using the Blur Tool 339
    Using the Sharpen Tool. 340
    Using the Smear Tool 341
    Using the Dodge Tool 342
    Using the Burn Tool 343
Drawing Shapes 343
    Using the Bezier Tool 344
    Using the B-Spline tool. 346
    Using the Ellipse and Rectangle Tools 347
Setting Default RotoPaint Tools and Settings 348
Selecting the Output Format and Channels 350
Selecting Existing Strokes/Shapes for Editing .351
    Viewing Point Numbers. 352
Editing Existing Stroke/Shape Attributes. 353
    Editing Attributes Common to Strokes and Shapes 353
    Transforming Strokes/Shapes/Groups 356
    Adjusting Mask Controls. 357
    Editing Shape Specific Attributes 358
    Editing Stroke Specific Attributes. 359
    Editing Clone or Reveal Attributes 362
Editing Existing Stroke/Shape Timing. 362
Editing Existing Stroke/Shape Stack Order 363

Editing Existing Stroke/Shape splines 363
Animating Strokes/Shapes 365
Copying, Pasting, and Cutting Stroke Positions.....368
    Copying Point Positions 368
    Pasting Point Positions.....369
    Cutting Point Positions.....369
RotoPaint and Stereoscopic Projects.....370
Where Are the Bezier and Paint Nodes 370

TEMPORAL OPERATIONS
Distorting Time.....371
    Simple Retiming.....371
    Interpolation.....372
    Frame-Blending.....373
    OFlow Retiming.....376
    OFlow Parameters.....376
    Warping Clips 379
    Global Frame Range and Speed.....381
Applying the TimeBlur Filter.....382
Editing Clips 382
    Slipping Clips.....382
    Cutting Clips 383
    Splicing Clips.....384

WARPING AND MORPHING IMAGES
Warping.....386
    Warping Images Using the GridWarp Node 386
    Warping an Image Using the SplineWarp Node 393
    Animating Warps.....398
Morphing.....400

CREATING EFFECTS
Background Reflections on Foreground Elements 406
Creating Star Filter Effects on Image Highlights 409
Creating Text Overlays.....412
    Creating a Text Overlay 412
    Repositioning and Transforming Text 416
    Fonts.....417
    Changing the Text Color.....420

ANALYZING FRAME SEQUENCES
Analysing and Matching Frame Sequences.....422

3D COMPOSITING
Overview.....426
Setting Up a Scene 427

  The Scene Node 427
  The ScanlineRender Node 428
  The Camera Node 428
Using the 3D Viewer. 429
3D Scene Geometry. 432
  Working with Cards. 432
  Working with Cubes 436
  Working with Spheres 437
  Working with OBJ Objects. 438
3D Selection Tools 439
  Matching Position, Orientation and Size to 3D Selection 441
  Parenting to Axis Objects 441
Merging Objects 443
Object Material Properties 443
Projecting Textures onto Objects. 447
  Projecting Textures with the UVProject Node 447
  Projecting Textures with the Project3D Node 448
Replacing Material Channels with a Constant Color. 449
Merging Shaders 450
  Merging Two Shader Nodes 450
  Merging a Material with the Objects Behind 452
Object Display Properties 456
Transforming Objects 457
  Using the Transform Handles 458
  Transforming from the Node Properties Panel. 459
  Transformations and the Pivot Point 460
  Using the TransformGeo Node 460
Modifying Object Shapes 463
  Modifying Objects Using Lookup Curves 463
  Modifying Objects Using a Power Function 464
  Modifying Objects Using an Image 466
  Modifying Objects Using a Perlin Noise Function. 467
  Modifying Objects Using a Distortion Function 468
  Modifying Objects Using a Trilinear Interpolation 469
Lighting 470
  Direct Light 470
  Point Light. 471
  Spot Light 472
  Environment Light 473
  The Light Node 475
Manipulating Object Normals 476
Working with Cameras 477

Projection Cameras 478
    First a Little Math... 478
    Setting Up the Projection Camera Script 479
Adding Motion Blur to the 3D Scene 481
Importing Channel Files, Cameras, Lights, Transforms, and Meshes from Other Applications 483
    Applying Tracks to an Object 483
    Working with FBX Files 484
    Importing Cameras from Boujou 490
    Exporting Geometry, Cameras, Lights, Axes, or Point Clouds 491
Rendering a 3D Scene 492
    Adjusting the Render Parameters 492

WORKING WITH STEREOSCOPIC PROJECTS

Setting Up Views for the Script 494
Loading Multi-View Images 496
Displaying Views in the Viewer 498
Selecting which Views to Apply Changes To 500
    Splitting Views Off 501
    Selecting the View to Process When Using the RotoPaint Node . . . 503
Performing Different Actions on Different Views 503
Reproducing Changes Made to One View. 504
    Reproducing Paint Strokes, Beziers, and B-spline Shapes 504
    Reproducing X and Y Values. 506
Swapping Views 507
Converting Images into Anaglyph 508
Changing Convergence 510
Previewing and Rendering Stereoscopic Images 515
    Flipbooking Stereo Images within FrameCycler 515
    Rendering Stereoscopic Images 515

PREVIEWS AND RENDERING

Previewing Output 517
    Previewing in a Nuke Viewer 517
    Flipbooking Sequences 518
    Previewing on an External Broadcast Video Monitor. 521
Rendering Output 522
    Render Resolution and Format 522
    Output (Write) Nodes 523
    File Name Conventions for Rendered Images 528
    Changing the Numbering of Rendered Frames 529
    Using a Write Node to Read in the Rendered Image 531
    Render Farms 532

EXPRESSIONS

Understanding Expression Syntax 533
    Linking Expressions. 533
    To Convert Expressions Between Scripting Languages 536
Adding Mathematical Functions to Expressions. 537

SETTING INTERFACE PREFERENCES

Displaying the Preferences Dialog 543
Changing Preferences 543
Saving Preferences 543
Resetting Preferences. 544
The Available Preference Settings 544
    Preferences Tab 545
    Windows Tab. 547
    Control Panels Tab 549
    Appearance Tab 550
    Node Colors Tab 551
    Node Graph Tab 552
    Viewers Tab 557
    Script Editor Tab. 559

THE SCRIPT EDITOR AND PYTHON

Workflow. 561
Using the Script Editor 562
Example Scripts 566
    Creating Nodes and Setting Their Parameters 566
    Assigning Variables. 568
    Adding Parameters to Nodes 568
    Connecting Nodes and Setting Their Inputs 570
    Setting Default Values for Controls 571
    Rendering with the Write Node 571
    Flipbooking with External Applications. 572
    Listing a Node's Controls 573
    Undoing and Redoing Actions. 573
    Frame Navigation 573
    Setting Frame Ranges Using Python. 574
    Copying an Animation Curve Between Nodes 577
    Overriding the Creation of a Particular Node. 577
    Getting Information on the Nuke Environment You Are Running. . .578
    Accessing Node Metadata 579
    Creating Dialogs and Panels 580
    Creating Progress Bar Dialogs 585
    Clearing Out the Current Nuke (.nk) Script 586
    Creating Views for a Stereoscopic Project. 586
    Adjusting Parameter Values in Stereo Projects 587
Automating Procedures 587

Getting Help 588
    More Documentation 588
    Viewing More Examples 589
    Using the Help Statement. 589
    Python on the Web 589
Python Callbacks Embedded in Scripts and Nodes. 589
    onUserCreate 590
    onCreate 591
    onScriptLoad. 591
    onScriptSave 592
    onScriptClose 592
    onDestroy 592
    knobChanged. 593
    updateUI 594
    autolabel 594
    beforeRender 595
    beforeFrameRender. 596
    afterFrameRender 596
    afterRender. 596
    filenameFilter 597
Using wxPython in Nuke to Write Custom GUIs. 597
    Installation 598
    General Workflow 599
Using PyQt4 in Nuke to Write Custom GUIs. 600
    Installation 600
    General Workflow 601
    Example. 601

CONFIGURING NUKE

What Is a Terminal and How Do I Use One?. 603
Command Line Operations 604
Environment Variables 611
    Setting Environment Variables 611
    Nuke Environment Variables. 614
Loading Gizmos, NDK Plug-ins, and TCL scripts. 616
Loading Python Scripts. 617
Loading OFX Plug-ins 617
Defining Common Favourite Directories. 618
Handling File Paths Cross Platform 621
Defining Custom Menus and Toolbars 622
Defining Common Image Formats. 629
Gizmos, Custom Plug-ins, and Generic TCL Scripts 629
    Creating and Sourcing Gizmos 630

    Custom Plug-ins 643
    Sourcing TCL Procedure 644
Template Scripts...644
Defining Common Preferences 645
Altering a Script's Lookup Tables (LUTs) 647
    Overview...647
    Displaying, Adding, Editing, and Deleting LUTs 648
    Selecting the LUT to Use 650
    Default LUT settings...................................650
    Example Cases 650
Creating Custom Viewer Processes 651
    Using a Gizmo as a Custom Viewer Process...............653
    Applying Custom Viewer Processes to Images.............657

NUKEX

NukeX Features 658
Installing NukeX...659
Licensing NukeX..659
Launching NukeX 659
    On Windows 659
    On Linux 660
    On Mac OS X..660

CAMERA TRACKING

Quick Start..661
Connecting the CameraTracker Node..........................662
Tracking Features in a Sequence 662
    Seeding Tracks 662
    Setting Tracking Parameters............................662
    Masking Out Regions of the Image.......................664
    Viewing Tracks and Track Information...................664
    Creating Manual Tracks 665
Setting the Camera Parameters..............................667
    Adjusting the Camera Parameters 667
Accounting for Lens Distortion.............................668
Solving the Camera Position................................669
    Clearing Automatic Tracks 670
Adjusting the Solve..670
    Deleting Tracks..670
    Refining the Solve.....................................671
Transforming the Scene 674
Adjusting the Virtual Camera...............................674

Centering on Selected Tracks 675
Troubleshooting the Solve 675
Using the Point Cloud. 675
Setting Points on Ground Origin or Different Axes 676
Setting a Ground Plane 676
Scaling a Scene 677
Attaching Objects to the Footage 677
Copying Translate and Rotate Values 678
Exporting a Point Cloud 678
Tracking Multiview Projects 678

ADDING AND REMOVING LENS DISTORTION

Quick Start 680
Calculating Lens Distortion Automatically 681
Image Analysis Parameters. 681
Analyzing Distortion Using a Grid 682
Grid Analysis Parameters 682
Analyzing Distortion Using Lines 682
Line Analysis Parameters 683
Adjusting LensDistortion Parameters. 683
Calculating the Distortion on One Image and Applying it to Another . . .685
Applying Lens Distortion to a Card Node. 685

CREATING DENSE POINT CLOUDS AND 3D MESHES

Quick Start 687
Connecting the PointCloudGenerator Node 687
Tracking a Dense Point Cloud. 688
Filtering Your Point Cloud. 689
Creating a Mesh Using a Point Cloud. 689
Adding Texture To The Point Cloud Mesh .690

GENERATING DEPTH MAPS

How is the Depth Calculated?. 692
Connecting DepthGenerator 693
To Connect the DepthGenerator Node 693
Generating a Depth Map. 693

USING THE MODELER NODE

Quick Start 696
Connecting The Modeler Node 696
Adding Faces. 696
Editing Faces and Vertices 698
Moving and Extruding Faces. 698
Adding Single Vertices 699
Adjusting Modeler Options. 699

USING PROJECTIONSOLVER

Quick Start 701
Connecting the ProjectionSolver Node 701
Matching 3D Geometry with 2D Footage 702
    Copying Values From a Tracker Node. 702
    Solving Your Camera 703
    Adjusting Solver and Lens Controls 703
Refining the Solve. 704
Creating Cards 705

RENDERING WITH PRMANRENDER

Setting Up RenderMan and PrmanRender 706
Using The PrmanRender Node 706
    Adjusting Render Quality 707
    Adjusting Shadows, Reflections, Refractions and Depth of Field. . .707
    Adjusting Motion Blur Parameters 708
    Arbitrary Output Variables Parameters 709
Using the Reflection Node 709
Using the Refraction Node 709

TUTORIALS

The Projects 711
Installing the Project Files 711

TUTORIAL 1: COMPOSITI NG BASICS

Starting Nuke 713
Using the Toolbar 715
Using the Menus. 715
Customizing Your Layout 716
Saving Files and File Backup. 718
Setting Up the Project 719
Working with Nodes 720
    Connection Tips 724
Importing Image Sequences 727
Navigating Inside the Windows 729
Working with Viewers. 731
Reformatting Images 735
Using Proxies and "Down-res" 736
Compositing Images 737
Color-Correcting Images. 740
Masking Effects 740
    To Create and Apply a Bezier Mask 741
Creating Flipbook Previews. 741

Rendering Final Output 742
Epilog 745

TUTORIAL 2: TRACKING, STABILISING, AND MATCHMOVING
One-Point, Two-Point, Three-Point, Four 747
Open the Tutorial Project File. 748
Tracking a Single Feature 748
Tracking Obscured Features. 752
Stabilising Elements 754
Matchmoving Elements. 757
Epilog 760

TUTORIAL 3: KEYING AND MATTES
Open the Tutorial Project File. 761
Keying with Primatte 763
Image-based Keying 768
Rotoscoping 773
Keying Video. 778
Epilog 786

TUTORIAL 4: 3D INTEGRATION
The Basic 3D System 788
    The 3D Viewer 788
    The Geometry or Scene Node 789
    The Camera Node 789
    The ScanlineRender Node 789
Open the Tutorial Project File. 790
Setting Up a 3D System 791
Making a Scene. 797
Merging and Constraining Objects 799
Animating a Scene 802
Working with Geometry 806
Lighting and Surface Properties 811
Epilog 814

APPENDICES

Organisation of the Section 815

APPENDIX A: HOTKEYS
Hotkeys. 816
Conventions 816
Node Graphs, Viewers, Curve Editors, Script Editors, and Properties Bins . . 817
Properties Panels 817
Node Graph. 818

Editing 820
Viewers 821
3D Viewer 824
RotoPaint Draw. 824
Curve Editor and Dope Sheet 826
Script Editor 827
Toolbar 827
Content Menus 827
Color Picker. 828

APPENDIX B: SUPPORTED FILE FORMATS
Supported File Formats. 829
Supported Image Formats. 829

APPENDIX C: CONVERTING FROM SHAKE TO NUKE
Converting from Shake to Nuke 832
Terms (and Conditions). 832
Node Reference 834
    Image Nodes 834
    Color Nodes 834
    Filter Nodes. 835
    Key Nodes 837
    Layer Nodes 837
    Other Nodes 838

APPENDIX D: THIRD PARTY LICENSES
Third Party Licenses 839

APPENDIX E: END USER LICENSING AGREEMENT
End User Licensing Agreement (EULA) 846

INDEX

A-Z 852

PREFACE

Nuke is an Academy Award® winning compositor. It has been used to create extraordinary images on scores of feature films, including *Avatar, District 9, Australia, The Dark Knight, Quantum of Solace, The Curious Case of Benjamin Button, Iron Man, Transformers, Pirates of the Caribbean: At World's End*, and countless commercials and music videos.

About this Manual

This manual consists of four sections:

1. Getting Started, which teaches you how to acquire a licence and install Nuke, use the interface and work with script files.
2. Reference, which describes key features of Nuke in more detail. You can dip in and out of this section depending on what you're interested in.
3. Tutorials, which are designed to show you how to solve common compositing problems and help you really learn Nuke.
4. Appendices, which include the available hotkeys, supported file formats, a Shake to Nuke conversion course, and the end user license agreement.

If you are new to Nuke, we recommend that you start by familiarizing yourself with the Getting Started section and working your way through the Tutorials. These two sections should give you a good base to build on when creating your own scripts. If you then need an answer to a specific problem or want to learn how to use a specific feature, you can always turn to the Reference section. All the sections are color-coded to make it easier for you to find what you are looking for.

Throughout this user guide, we assume you have a basic knowledge of computer graphics and digital compositing theory, as well as proficiency with the operating system for which Nuke is installed.

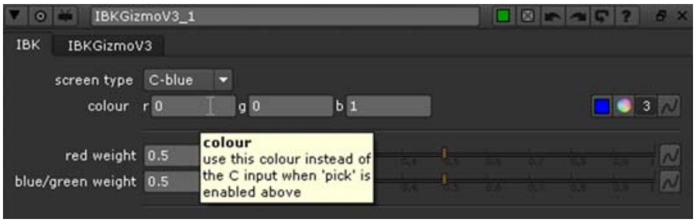
Note *For the most up-to-date information, please see the Nuke product page and the latest Nuke user guide on our web site at www.thefoundry.co.uk.*

Getting Help

Viewing Online Help

Nuke features several forms of online help:

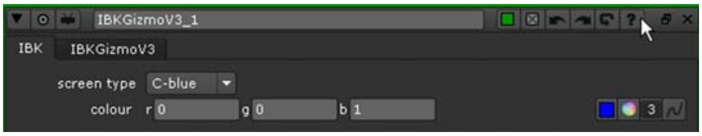
- Most controls offer concise instructions in the form of tool tips. To display the tool tips, move your mouse pointer over an interface control or a node parameter.



screen type C-blue
colour r 0 g 0 b 1
red weight 0.5
blue/green weight 0.5

colour
use this colour instead of the C input when 'pick' is enabled above

- Many properties panels include contextual descriptions of the node's parameters. To display these descriptions, click the **?** icon.

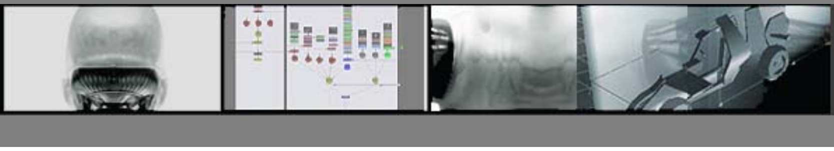


screen type C-blue
colour r 0 g 0 b 1

- Finally, you can click the **Help** menu to access the following:
  - **Key assignments** - a list of hotkeys.
  - **Documentation** - this user guide, the Nuke Developer Kit (NDK), and documentation for using FrameCycler, Python, TCL, and expressions in Nuke.
  - **Training and tutorials** - FX PHD's Compositor's Guide to Nuke training videos, and a list of other training resources. You can also find grain samples, and the files used with the tutorials in this user guide.
  - **Mailing lists** - information on Nuke-related e-mail lists.
  - **Plug-in Installer** - download plug-ins for Nuke.

Contacting Customer Support

Should questions arise that this manual or the online help system fails to address, you can contact Customer Support directly via e-mail at support@thefoundry.co.uk or via telephone to our London office on +44 (0)20 7434 0449 or to our Los Angeles office on (310) 399 4555 during office hours.



GETTING STARTED

By now, you are probably itching to dive into and play with Nuke, so let's get you started. This section provides you with all you need to start compositing with Nuke. For more detailed information on Nuke and its functions, see the "Reference" section on page 160 in this manual.

About the Chapters

Before you can start exploring Nuke's wonders, you of course need to install Nuke on your machine. For instructions on how to do so and launch either the commercial version of Nuke or the Nuke Personal Learning Edition, refer to Chapter 1: "Installation and Licensing" on page 25.

After the installation, you'll come across the Nuke Plug-in Installer. With this application, you can easily get information on plug-ins you can use with Nuke and download them. For more information on the Nuke Plug-in Installer, go to Chapter 2: "The Nuke Plug-in Installer", on page 38.

Once you have successfully installed and launched Nuke, you can sit back and start familiarizing yourself with the interface. Chapter 3: "Using the Interface" on page 39 is designed to help you understand the workflow, the workspace and the different controls. It also provides you with information on adjusting the interface to suit your preferences.

Finally, to learn about scripts and script management, turn to Chapter 4: "Managing Scripts" on page 121.

1 INSTALLATION AND LICENSING

We know the installation and licensing of a new application can be a boring task that you just want to be done with as soon as possible. To help you with that, this chapter guides you to the point where you have an open workspace of Nuke in front of you and are ready to start compositing to your heart's content, whether it be with the commercial version of Nuke or the Nuke Personal Learning Edition (PLE).

System Requirements

Before you do anything else, check the system requirements to make sure your computer can run Nuke.

Windows and Linux

- 550 MHZ Pentium III or newer processor
- Windows XP (with Service Pack 2 or later), Windows 7 (64-bit), or Linux RHEL 5.4
- 5 GB disk space available for caching and temporary files
- 512 MB RAM (minimum requirement)
- Workstation-class graphics card, such as NVIDIA Quadro series, ATI FireGL series, R3D Rocket, or newer. Driver support for OpenGL 2.0.

  To enable optional GPU acceleration of certain effects, you need OpenGL 2.0 with support for floating point textures and GLSL
- Display with at least 1280 x 1024 pixel resolution and 24-bit color
- Three-button mouse.

Note *To avoid graphical problems, such as text disappearing in the Viewer and Node Graph, it is important to keep your graphics card drivers up-to-date. Driver updates can be obtained from the web sites of the graphics card manufacturers (for example, www.nvidia.com and www.ati.com).*

Note *If you're using R3D Rocket graphics card, note that using it in Nuke will most likely only be considerably faster when you're reading in at full resolution. If you're reading in at half resolution for instance, using Nuke without the R3D Rocket card enabled may be faster. This is because the R3D Rocket graphics card is designed to be fast when reading in multiple frames at the same time. This is not how Nuke works internally, and therefore reads with the R3D Rocket card disabled will sometimes be faster when working in lower resolutions (< 4K widths). Note that the R3D Rocket card will always produce better results when downsampling than Nuke will.*

Also, the R3D Rocket card can only be used by one application at a time, so if you are viewing multiple Nuke scripts at once, you will only be able to use the R3D Rocket card in one.

Mac OS X

- Intel processor and Mac OS X 10.5 "Leopard" or 10.6 "Snow Leopard"
- For Nuke 64-bit Mac support, you will need a Mac with an Intel Core 2 Duo or later.
- 5 GB of disk space available for caching and temporary files
- 512 MB of RAM (minimum requirement)
- AGP, PCI Express, or R3D Rocket graphics card with at least 32 MB of video memory. Driver support for OpenGL 2.0.
- Display with at least 1280 x 1024 pixel resolution and 24-bit color
- Three-button mouse

Licensing Nuke

If you simply want to try out or learn Nuke, you can run the Nuke Personal Learning Edition (PLE) without a license key. The PLE allows you to explore practically all Nuke's features, but prevents the commercial use of the application. To use the PLE, you only need to install Nuke on your machine (see "Installing Nuke" on page 29) and launch it in a special way described in "Launching the Nuke Personal Learning Edition (PLE)" on page 35. "About the Personal Learning Edition" on page 36 also provides you with more information on the PLE and how it differs from the commercial version of Nuke.

To use the commercial version of Nuke, you need a valid license key. You can obtain one by purchasing Nuke or by requesting a time-limited demo license. The instructions below tell you how to get a license key and what to do with it. Without a valid license key, the commercial version of Nuke will fail to run.

Nuke uses FLEXlm encryption in the license keys. Node-locked (uncounted) and floating (counted) licenses are supported. We also supply a suite of tools to manage and monitor floating licenses running on a server across a network of machines. These tools are called Foundry FLEXlm Tools (FFT) and can be downloaded free of charge from our web site at www.thefoundry.co.uk/licensing. The Foundry FLEXlm Tools should be installed on the server.

Obtaining Licenses

You can request licenses by contacting The Foundry Sales Department (sales@thefoundry.co.uk). To generate you a license key, we need to know your System ID. The System ID (sometimes called lmhostid) returns a unique number for your computer. We lock our license keys to the System ID.

To display your System ID, do any of the following:

- Download the Foundry System ID (FSID) utility from www.thefoundry.co.uk/licensing and run it. Your System ID will be displayed in the window that opens.
- Install and try to launch the commercial version Nuke without a valid license (see "Installing Nuke" on page 29 and "Launching Nuke" on page 33). Your System ID number is displayed in the license error you get.
- Download the Foundry FLEXlm Tools (FFT) free of charge from our web site and then run the following command in a terminal shell:

  ```
  /usr/local/foundry/FLEXlmTools5.0/bin/lmutil lmhostid
  ```

Just so you know what a System ID number looks like, here's an example: 000ea641d7a1.

Once you have provided us with your System ID number and a license key has been generated for you, you will receive the license key in an e-mail or Internet download. The license key is contained in a text file called *foundry.lic*.

Here is an example node-locked (uncounted) license for Nuke that expires on 29 September 2010 for a computer running Nuke with a System ID of 000a957bfde5. Node-locked licenses allow you to run Nuke on one machine only.

```
INCREMENT nuke_i foundry 2009.0929 29-sep-2010 uncounted \
HOSTID=000a957bfde5 ISSUED=29-sep-2009 \
SIGN="00DA 99A9 E744 217E 8AD3 E7AF E289 C0C6 \
6B23 2891 AC01 0F50 E64D 8847 8B22 3A40 2BE9 \
A268 B7C2 4BC0 36AF"
INCREMENT nuke_r foundry 2009.0929 29-sep-2010 uncounted \
HOSTID=000a957bfde5 ISSUED=29-sep-2009 \
SIGN="03C9 1D0D 5503 EC34 2CAF 37C0 8731 5E57 \
06E8 C8CB E113 51EA 87C6 3BE8 242B 50AC 35EE \
6753 B3AB 3AC4 1559"
```

And here's an example of a permanent floating license for a server whose machine name is "red" with System ID 000ea641d7a1 communicating on port number 30001 that will enable any version of Nuke built before 31 July 2009 to run on up to 5 machines simultaneously.

```
SERVER red 000ea641d7a1 30001
VENDOR foundry
```

```
INCREMENT nuke_i foundry 2009.0731 permanent 5 ISSUED=15-
jul-2009 SIGN="00DA 99A9 E744 217E 8AD3 E7AF
E289 C0C6 6B23 2891 AC01 0F50 E64D 8847 8B22
3A40 2BE9 A268 B7C2 4BC0 36AF"
INCREMENT nuke_r foundry 2009.0731 permanent 5 ISSUED=15-
jul-2009 START=31-jul-2009 SIGN="03C9 1D0D 5503
EC34 2CAF 37C0 8731 5E57 06E8 C8CB E113 51EA
87C6 3BE8 242B 50AC 35EE 6753 B3AB 3AC4 1559"
```

For information on what to do with the foundry.lic file, see "Installing Licenses" below.

Installing Licenses

Once a license has been created for you, you will receive your license key (foundry.lic) in an e-mail or Internet download. You should also receive the Foundry License Installer (FLI) application to help you install the license key. The FLI is also available to download from www.thefoundry.co.uk/licensing. The instructions below tell you what to do with these.

To install a license:

1. Make sure you have saved both the license key (foundry.lic) and the Foundry License Installer application in the same directory. You should receive these by e-mail or (if you purchased Nuke online) in a download.
2. Double-click on the Foundry License Installer application to run it. The license key should automatically appear in the FLI window, if the FLI and foundry.lic are in the same directory. If they are not, you can either copy and paste the contents of the license key or drag and drop the file into the FLI window.
3. Click **Install**. Provided that the license is valid, the license will be installed to the correct directory.

   If you installed a node-locked license key (a license that allows you to run Nuke on one machine only), you're good to go. Proceed to "Launching Nuke" on page 33.
4. If you have a floating license, you will be asked whether you want to create a client license file. You should accept this and save the file (client.lic). Copy the client.lic file to each machine that you wish to run Nuke on and use the FLI to install it on them. In order for the floating license to work, you will need to install the Foundry FLEXlm Tools 5.0 (FFT) on the license server machine. For more information on how to install floating licenses, refer to the FFT user guide, which you can download from our web site.

Further Reading

There is a lot to learn about licenses, much of which is beyond the scope of this manual. For more information on licensing Nuke, displaying the System ID number, setting up a floating license server, adding new license keys and managing license usage across a network, you should read the Foundry FLEXlm Tools User Guide, which can be downloaded from our web site, www.thefoundry.co.uk/licensing.

Installing Nuke

Nuke 6.2 will install separately to any previous versions installed. Nuke 6.2 is available to download from our web site at www.thefoundry.co.uk. The downloads are in compressed tar format (tgz) for Linux, exe for Windows, and dmg format for Mac OS X. The installation procedure is the same for both the Nuke Personal Learning Edition and the commercial version of Nuke.

Installation on Windows

To install Nuke on Windows, do the following:

1. Download the correct installation file from our web site. Which installation file you download depends on whether you are using a 32- or a 64-bit machine.
2. Double-click on the file to install Nuke.
3. Follow the on-screen instructions. By default, Nuke is installed to *drive letter*:\Program Files\Nuke 6.2v2 (unless you're installing 32-bit Nuke on 64-bit Windows, in which case the default location is *drive letter*:\Program Files (x86)\Nuke 6.2v2).
4. Proceed to "Launching Nuke" on page 33.

Note *On Windows, if you install Nuke to a network drive to run from multiple computers, please ensure that the correct Microsoft run time libraries are installed on each machine that will run Nuke. To do this, run "vcredist_x86.exe" (32-bit) or "vcredist_x64.exe" (64-bit) on each machine. The appropriate one of these files can be found in the "VCRedist" subdirectory in the folder where Nuke is installed - for example, vcredist_x86.exe for the 32-bit version of Nuke.*

Running Nuke without installing the libraries on your machine may work correctly, particularly as many systems (such as Windows Vista by default) will already have them. If the libraries are not present, Nuke will still run correctly, but some plug-ins may fail to load with error messages such as "This application has failed to start because the application configuration is incorrect. Reinstalling the application may fix this problem."

Please note that these libraries are set up automatically on the machine that runs the Nuke installer, so users installing on their local machine will not need to worry about this issue.

Installing Nuke from the command line

To install Nuke from the command line, do the following:

1. Download the correct .exe installation file from our web site at www.thefoundry.co.uk. Which installation file you download depends on whether you are using a 32- or a 64-bit machine.
2. To open a command prompt window, select **Start** > **All Programs** > **Accessories** > **Command Prompt**.
3. Use the **cd** (change directory) command to move to the directory where you saved the installation file. For example, if you saved the installation file in **C:\Temp**, use the following command and press **Return**:
   ```
   cd \Temp
   ```
4. To install Nuke, do one of the following:
   - To install Nuke and display the installation dialog, type the name of the install file without the file extension and press **Return**:
     ```
     Nuke6.2v16.2v2-win-x86-release-32 (on a 32-bit machine)
     Nuke6.2v2-win-x86-release-64 (on a 64-bit machine)
     ```
   - To install Nuke silently so that the installer does not prompt you for anything but displays a progress bar, enter **/silent** after the installation command:
     ```
     Nuke6.2v16.2v2-win-x86-release-32 /silent
     Nuke6.2v2-win-x86-release-64 /silent
     ```
   - To install Nuke silently so that nothing is displayed, enter **/verysilent** after the installation command:
     ```
     Nuke6.2v16.2v2-win-x86-release-32 /verysilent
     Nuke6.2v2-win-x86-release-64 /verysilent
     ```

Note *By running a silent install of Nuke, you agree to the terms of the End User Licensing Agreement. To see this agreement, please refer to "Appendix E: End User Licensing Agreement" on page 846 or run the installer in standard non-silent mode.*

Installation on Linux

To install Nuke on Linux, do the following:

1. Download the correct .tgz installation file from our web site at www.thefoundry.co.uk.
2. Extract the installer from the tgz archive with the following terminal command:

```
tar xvzf Nuke6.2v2-linux-x86-release-64.tgz
```
(on a 64-bit machine)

This will give you an installer file.

3. Run the installer.
   ```
   sudo ./Nuke6.2v2-linux-x86-release-64-installer
   ```
   (on a 64-bit machine)
4. Follow the on-screen instructions. By default, Nuke is installed to /usr/local/Nuke`6.2v2`.
5. If you didn't add a license key during the installation, do that now. Proceed to "Launching Nuke" on page 33.

Tip *To install Nuke silently, you can simply unzip the installer file. This creates the properly formed Nuke directory tree in the current directory.*

By installing Nuke silently, you agree to the terms of the End User Licensing Agreement. To see this agreement, please refer to "Appendix E: End User Licensing Agreement" on page 846 or run the installer in standard non-silent mode.

Installing Nuke remotely from the command line

If you need to install Nuke on render machines using the command line, do the following:

1. Download the correct .tgz installation file from our web site at www.thefoundry.co.uk. Which installation file you download depends on whether you are using a 32- or a 64-bit machine.
2. Extract the installer from the tgz archive with the following terminal command:
   ```
   tar xvzf Nuke6.2v2-linux-x86-release-64.tgz
   ```
   (on a 64-bit machine)

   This will give you an installer file.
3. Use the following terminal command to log in to your render machine as root:
   ```
   ssh root@render_machine
   ```
   Replace render_machine with the name of your render node.
4. Make a directory to install Nuke to:
   ```
   mkdir /usr/local/Nuke6.2v2
   ```
5. Copy the installer file from the machine that you downloaded it on to your render machine with a command like:
   ```
   scp root@download_machine:/tmp/Nuke6.2v2-linux-x86-release-64-installer root@render_machine:/usr/local/Nuke6.2v2/
   ```
   (on a 64-bit machine)

Replace *download_machine* with the name of the machine you downloaded the installer file to, and *render_machine* with the name of your render node.

6. Unzip the installer file to unpack its contents into your Nuke directory:
   ```
   cd /usr/local/Nuke6.2v2
   unzip Nuke6.2v2-linux-x86-release-64-installer
   ```
   (on a 64-bit machine)
7. Repeat steps 3-6 for each render machine.

Installation on Mac OS X

To install Nuke on Mac OS X, do the following:

1. Download the correct .dmg installation file from our web site at www.thefoundry.co.uk. Which installation file you download depends on whether you are using a 32- or a 64-bit machine.
2. Double-click on a dmg archive to extract the pkg installer:
   - Nuke6.2v16.2v2-mac-x86-release-32.dmg (for 32-bit)
   - Nuke6.2v2-mac-x86-release-64.dmg (for 64-bit)

   A pkg file is created.
3. Double-click on the pkg file.
4. Follow the on-screen instructions to install Nuke. By default, Nuke is installed to

   /Applications/Nuke6.2v16.2v2-32 (for 32-bit) or

   /Applications/Nuke6.2v2 (for 64-bit).
5. Proceed to "Launching Nuke" on page 33.

Installing Nuke silently from the command line

1. Download the .dmg installation file from our web site at www.thefoundry.co.uk.
2. Launch a Terminal window.
3. To mount the dmg installation file, use the **hdiutil attach** command with the directory where you saved the installation file. For example, if you saved the installation file in **Builds/Nuke**, use the following command:
   ```
   hdiutil attach /Builds/Nuke/Nuke6.2v16.2v2-mac-x86-release-32.dmg
   ```
   (on a 32-bit machine)
   ```
   hdiutil attach /Builds/Nuke/Nuke6.2v2-mac-x86-release-64.dmg
   ```
   (on a 64-bit machine)
4. Enter the following command:
   ```
   pushd /Volumes/Nuke6.2v2/
   ```
   This stores the directory path in memory, so it can be returned to later.
5. To install Nuke, use the following command:

```
sudo installer -pkg Nuke6.2v16.2v2-mac-x86-release-
32.pkg -target "/"
```
(on a 32-bit machine)

```
sudo installer -pkg Nuke6.2v2-mac-x86-release-64.pkg -
target "/"
```
(on a 64-bit machine)

You are prompted for a password.

6. Enter the following command:

```
popd
```

This changes to the directory stored by the pushd command.

7. Finally, use the following command to eject the mounted disk image:

```
hdiutil detach /Volumes/Nuke6.2v2
```

Note *By running a silent install of Nuke, you agree to the terms of the End User Licensing Agreement. To see this agreement, please refer to "Appendix E: End User Licensing Agreement" on page 846 or run the installer in standard non-silent mode.*

Launching Nuke

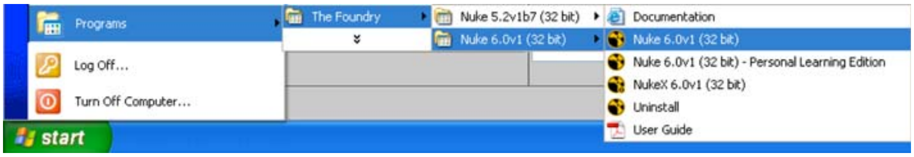
With the installation and licensing out of the way, you're ready to start compositing with Nuke. Depending on whether you want to use the Nuke Personal Learning Edition or the commercial version of Nuke, the procedure for launching Nuke is slightly different. Both are described below, beginning with the procedure for the commercial version.

Launching the Commercial Version

To launch the commercial version of Nuke, do one of the following:

On Windows

- Double-click the Nuke icon on the Desktop.
- Select **Nuke6.2v2 (32/64 bit)** from **Start** > **All Programs** > **The Foundry > Nuke6.2v2 (32/64 bit)**.

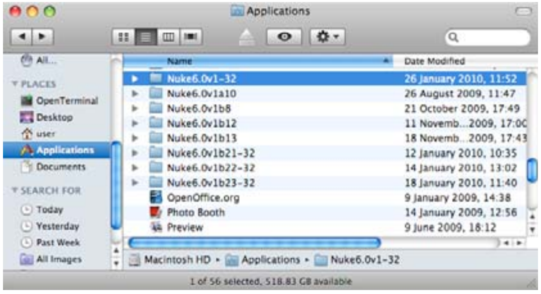

Programs
Log Off...
Turn Off Computer...
start
The Foundry
Nuke 5.2v1b7 (32 bit)
Nuke 6.0v1 (32 bit)
Documentation
Nuke 6.0v1 (32 bit)
Nuke 6.0v1 (32 bit) - Personal Learning Edition
NukeX 6.0v1 (32 bit)
Uninstall
User Guide

- Using a command prompt, navigate to the Nuke application directory (by default, **\Program Files\Nuke6.2v2** or \Program Files (x86)\Nuke6.2v2), and enter **nuke6.2**.

The Nuke graphical interface appears.

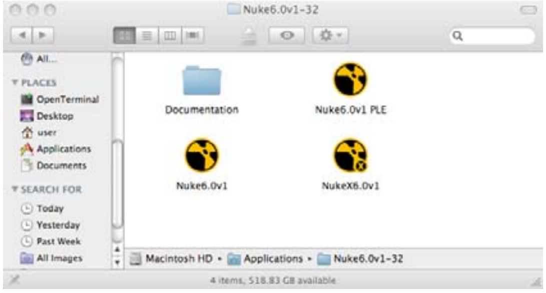
On Mac OS X

- Click the Nuke dock icon.
- Open the Nuke application directory (by default, **/Applications/Nuke**6.2v2**(32/64 bit)/**), and double-click the **Nuke** icon (or list item).



Name	Date Modified
Nuke6.0v1-32	26 January 2010, 11:52
Nuke6.0v1a10	26 August 2009, 11:47
Nuke6.0v1b8	21 October 2009, 17:49
Nuke6.0v1b12	11 Novemb...2009, 17:0
Nuke6.0v1b13	18 Novemb...2009, 17:43
Nuke6.0v1b21-32	12 January 2010, 10:35
Nuke6.0v1b22-32	14 January 2010, 13:02
Nuke6.0v1b23-32	18 January 2010, 11:40
OpenOffice.org	9 January 2009, 14:38
Photo Booth	14 January 2009, 12:56
Preview	9 June 2009, 18:12

Applications
All...
PLACES
OpenTerminal
Desktop
user
Applications
Documents
SEARCH FOR
Today
Yesterday
Past Week
All Images
Macintosh HD ▸ Applications ▸ Nuke6.0v1-32
1 of 56 selected, 518.83 GB available



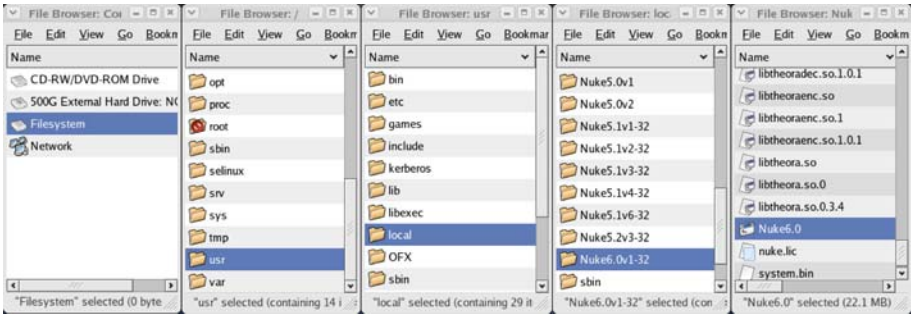
Nuke6.0v1-32
All...
PLACES
OpenTerminal
Desktop
user
Applications
Documents
SEARCH FOR
Today
Yesterday
Past Week
All Images
Documentation
Nuke6.0v1 PLE
Nuke6.0v1
NukeX6.0v1
Macintosh HD ▸ Applications ▸ Nuke6.0v1-32
4 items, 518.83 GB available

- Open a terminal, navigate to **/Applications/Nuke**6.2v2**-(32/64)/Nuke**6.2v2**.app/Contents/MacOS**, and enter **./Nuke**6.2.

The Nuke graphical interface displays.

On Linux

- Double-click the Nuke icon on the Desktop.
- Open the Nuke application directory (by default, **/usr/local/Nuke**6.2v2**/**) and double-click the **Nuke** icon (or list item).


File Browser: Co
File Edit View Go Bookm
Name
CD-RW/DVD-ROM Drive
500G External Hard Drive: N
Filesystem
Network
"Filesystem" selected (0 byte
File Browser: /
opt
proc
root
sbin
selinux
srv
sys
tmp
usr
var
"usr" selected (containing 14 i
File Browser: usr
bin
etc
games
include
kerberos
lib
libexec
local
OFX
sbin
"local" selected (containing 29 it
File Browser: loc
Nuke5.0v1
Nuke5.0v2
Nuke5.1v1-32
Nuke5.1v2-32
Nuke5.1v3-32
Nuke5.1v4-32
Nuke5.1v6-32
Nuke5.2v3-32
Nuke6.0v1-32
sbin
"Nuke6.0v1-32" selected (con
File Browser: Nuk
libtheoradec.so.1.0.1
libtheoraenc.so
libtheoraenc.so.1
libtheoraenc.so.1.0.1
libtheora.so
libtheora.so.0
libtheora.so.0.3.4
Nuke6.0
nuke.lic
system.bin
"Nuke6.0" selected (22.1 MB)

- Open a terminal, navigate to the Nuke application directory (by default, **/usr/local/Nuke**), and enter **./Nuke**6.2.

The Nuke graphical interface displays.

Launching the Nuke Personal Learning Edition (PLE)

You launch the PLE from a terminal (a window where you can enter commands directly rather than making selections through a user interface).

To launch the PLE:

- On Windows: Select the Personal learning Edition from **Start** > **All Programs** > **The Foundry** > **Nuke**6.2v2.

  OR

  Using a command prompt, navigate to the Nuke application directory (by default, /Program Files/Nuke6.2v2), and enter **nuke6.2 --ple**.
- **On Linux**: Open a terminal, navigate to the Nuke application directory (by default, /usr/local/Nuke), and enter **./Nuke**6.2 **--ple**.
- **On Mac OS X**: Click the Nuke PLE dock icon.

  OR

  Open the Nuke application directory (by default, **/Applications/ Nuke**6.2v2**-(32/64)/**), and double-click the **Nuke**6.2v2 **PLE** icon (or list item).

  OR

  Open a terminal, navigate to /Applications/Nuke6.2v2/Nuke6.2v2.app/ Contents/MacOS, and enter **./Nuke**6.2 **--ple**.

Note *On Mac OS X, you shouldn't move the Nuke PLE bundle from its original installation folder as this will prevent it from working correctly.*

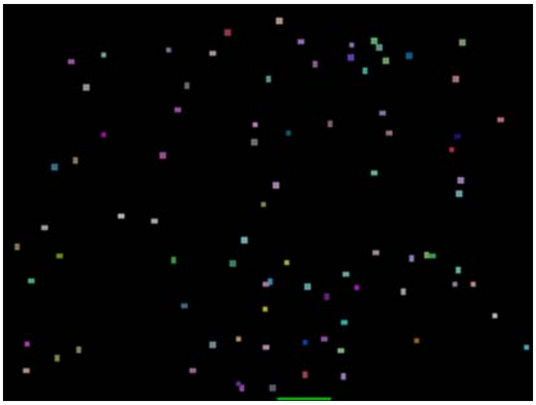
About the Personal Learning Edition

The Nuke Personal Learning Edition is a special version of Nuke that you can run without a license. The Personal Learning Edition is meant for personal, educational, and other non-commercial use. It is aimed at students, industry professionals, and others interested in Nuke. It includes all the features of the commercial version of Nuke, offering you a chance to explore and learn the application fully while using it from the comfort of your own home.

PLE Versus the Commercial Version of Nuke

The PLE is a fully functional version of Nuke, but, being aimed for non-commercial use only, it does differ from the commercial version in some aspects. Here are the main differences:

- **Watermark**. The PLE displays a watermark (shown below) on any images in the Viewer as well as images rendered out to files. This is to prevent the commercial use of the images.



- **External data storage**. All external data storage is encrypted in the PLE, including Nuke scripts (these are saved with the extension .nkple), gizmos (saved with the extension .gzple), and copying to the clipboard. Among other things, this means the PLE saves files in an encrypted format, unlike the commercial version of Nuke, which saves scripts unencrypted as plain text. The commercial version of Nuke cannot load files created with the PLE.

  The PLE, however, can load scripts and gizmos created with the commercial version.
- **Scripting**. In PLE mode, Nuke restricts the amount of nodes that can be retrieved at a time by scripting. Functions such as "nuke.allNodes()" in Python will return only the first 10 nodes available rather than all of them at once, and scripts written to iterate through the Node Graph will not be able to retrieve any more nodes beyond a set point. The commercial version of Nuke can retrieve any and all nodes at any time as the command names would suggest.

- **WriteGeo**. The WriteGeo node is disabled in the PLE.
- **Primatte**. The Primatte node is disabled in the PLE.
- **FrameCycler**. FrameCycler is disabled in the PLE.
- **Monitor output**. There is no video monitor output support in the PLE.
- **Plug-ins**. Only plug-ins that are shipped with Nuke, such as OFlow, can be used in the PLE. OFX plug-ins (such as The Foundry's Furnace) and custom plug-ins compiled with the NDK can only be used in the commercial version of Nuke.
- **Command line rendering**. It is not possible to render a PLE script with -x from the command line.

In other respects, the PLE contains all the functionality of the commercial version of Nuke.

2 THE NUKE PLUG-IN INSTALLER

This chapter gives you an overview of The Nuke Plug-in Installer application.

What Is the Nuke Plug-in Installer

The Nuke Plug-in Installer is an application in Nuke that gives you easy access to a large selection of useful plug-ins to use with Nuke.

Accessing the Nuke Plug-in Installer

The Nuke Plug-in Installer application is installed with Nuke, so you don't have to download or install it separately. The application will open up after you have installed Nuke.

Getting an Overview of the Plug-ins

On the **Plug-ins** tab of the Nuke Plug-in Installer, you can have a look at the selection of plug-ins available through the application.

Viewing Details of Plug-ins

To have a closer look at any of the plug-ins, just double-click on the icon and move on to the **Details** tab. You'll see a HTML page with more detailed information on the plug-in. At the bottom of the **Details** tab, you can select your operating system and download the plug-in software by clicking the **Download selected platform** button.

Downloading Plug-ins

When you've chosen a plug-in you want to download, do the following:

1. Double-click it on the **Plug-ins** tab. This will open the Plug-in information in the **Details** tab.
2. Select the operating system(s) you require using the checkboxes at the bottom of the window.
3. Click the **Download selected platforms** button.

   Nuke Plug-in Installer will prompt you to choose a download destination for the plug-in and then automatically start downloading it for you.
4. You'll see a notification when the download is finished, and after that you can install your plug-in by clicking the **Install** button in the Plug-in Installer.

3 USING THE INTERFACE

This chapter is designed to help you understand Nuke's workflow, learn how to use the interface, and customize the interface to suit your preferences.

Understanding the Workflow

Nuke utilizes a node-based work flow, where you connect a series of nodes to read, process, and manipulate images. Each node in the project—an image keyer, a color-correction, or a blur filter, for example—performs an operation and contributes to the output.

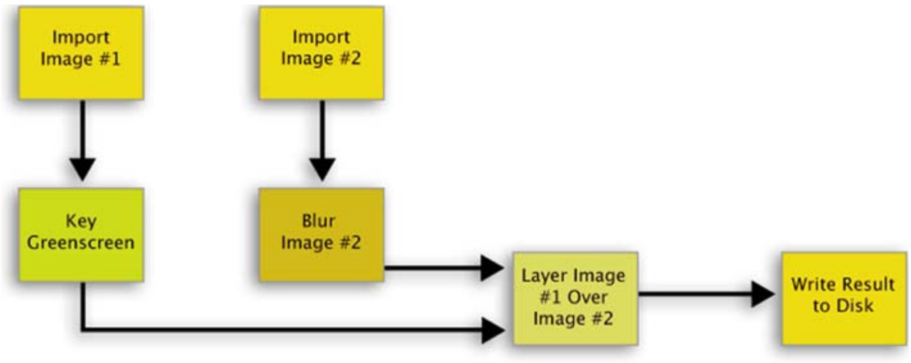

Import Image #1
Import Image #2
Key Greenscreen
Blur Image #2
Layer Image #1 Over Image #2
Write Result to Disk

Figure 3.1: A Nuke project consists of a network of linked operators called *nodes*.

Saved projects are called *script files*. You can open a Nuke project file in a text editor, and you will see a series of sequential commands which are interpreted and executed when you render the output.

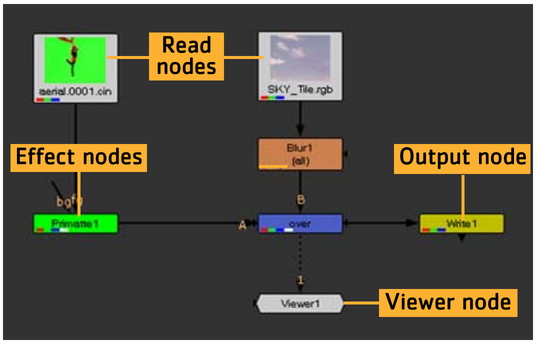

Read nodes
serial.0001.cin
SKY_Tile.rgb
Effect nodes
Blur1 (all)
Output node
bg
Primatte1
A
over
B
Write1
1
Viewer1
Viewer node

Figure 3.2: A simple Nuke script.

In Figure 3.2, you see an example of a very simple Nuke script. Two *Read* nodes reference image sequences on disk. Effect nodes extract a matte and blur an image. A merge node (named *over*) composites the foreground image over the background. Finally, a *Write* node renders and outputs the completed composite to disk. You'll also see a *Viewer* node, which displays the output of any node in the script.

The Nuke Window

The Nuke interface is constructed using interchangeable panes, panels, and tabbed panels so that you can customize your workspace to suit your particular needs.

Panes and Panels

Nuke's main window is divided into three panes: the Node Graph/Curve Editor pane, the Properties/Script Editor pane, and the Viewer pane.

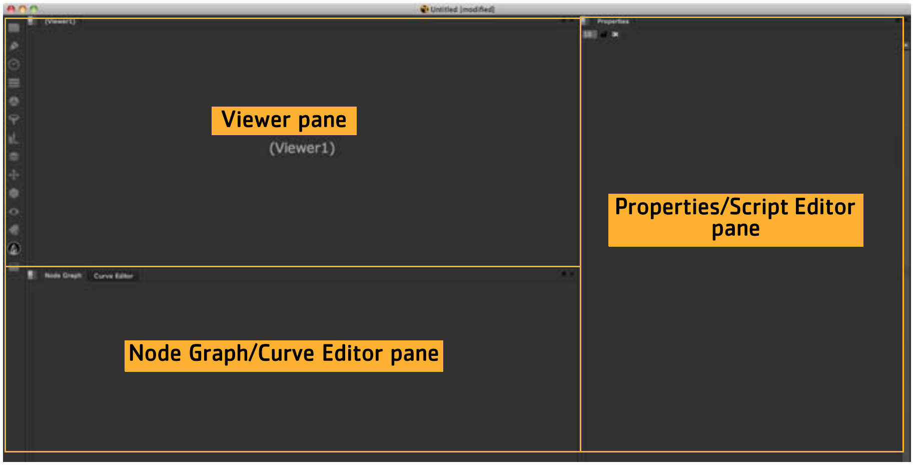

Untitled (modified)
Viewer1
Viewer pane
(Viewer1)
Properties
Properties/Script Editor pane
Node Graph
Curve Editor
Node Graph/Curve Editor pane

Figure 3.3: Empty main window.

Onto these panes, you can add the following panels:

- Toolbars for selecting nodes
- Node Graphs (also known as DAGs) for building node trees
- Curve Editors for editing animation curves
- Properties Bins for adjusting the nodes' controls
- Viewers for previewing the output
- Progress panels for displaying progress bars
- Script Editors for executing Python commands.

By default, there is a Node Graph panel in the lower left corner, a Viewer panel in the top left corner, and a Properties Bin on the right, as shown in Figure 3.4.

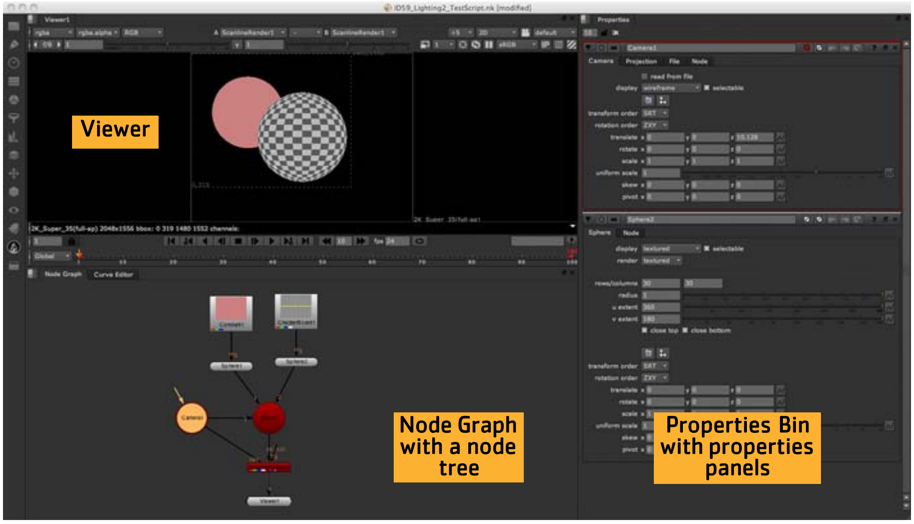

Viewer
Node Graph with a node tree
Properties Bin with properties panels

Figure 3.4: Main window with a Node Graph, Viewer, and Properties Bin.

The Node Graph is where you add nodes and build your node tree. When you add a node to the panel, its properties panel appears in the Properties Bin on the right. This is where you can adjust the node to produce the effect you're after. To check the result, you can view the output in a Viewer.

You can open more panels using the content menus described in "Toolbar, Menu Bar, and Content Menus" below. You can add several panels on the same pane and switch between them by using the tabbed pages on top of the pane.

Tabbed Panels

Panes are divided into tabbed panels on the top of the pane.

To go to a different tab, simply click on the tab name.

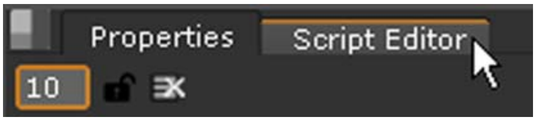


Figure 3.5: Moving from the Properties tab to the Script Editor tab.

To cycle through tabs, press **Ctrl/Cmd+T**.

Toolbar, Menu Bar, and Content Menus

The Toolbar is located on the left-hand side of the Nuke window. By default, it consists of twelve icons. The different nodes are grouped under these icons based on their functions. You use the Toolbar to add nodes to the Node Graph.

The menu bar is located on top of the Nuke window. Its menus, such as the **File** or **Edit** menu, let you perform more general actions related to the whole script, the Viewers, or editing, rather than certain individual nodes.

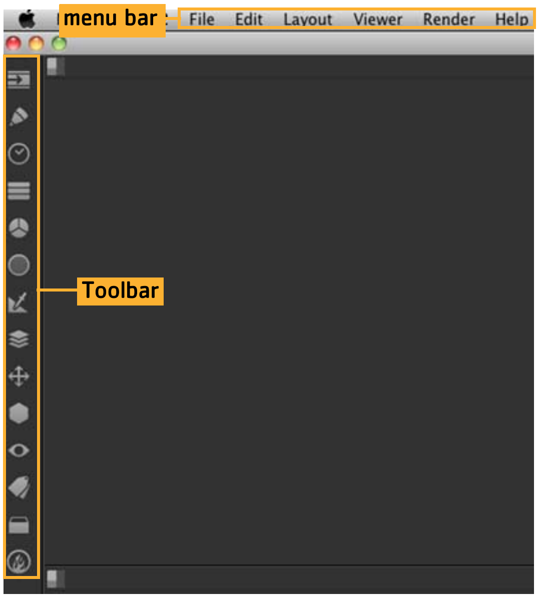


Figure 3.6: The Toolbar and the menu bar.

In addition to the Toolbar and the menu bar, you should also familiarize yourself with the content menus. They are the gray checkered boxes in the top left corner of each pane. If you click on the box, a pop-up menu opens. You can use the options in the pop-up menu to customize the window layout.

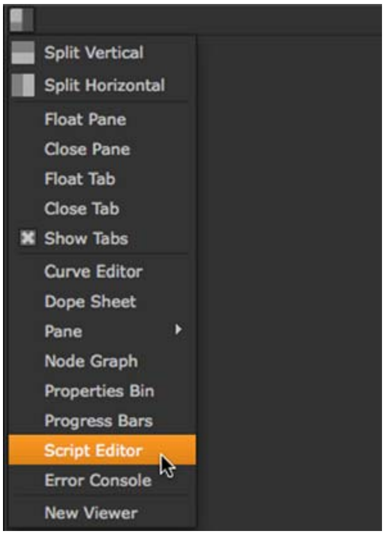

Split Vertical
Split Horizontal
Float Pane
Close Pane
Float Tab
Close Tab
Show Tabs
Curve Editor
Dope Sheet
Pane
Node Graph
Properties Bin
Progress Bars
Script Editor
Error Console
New Viewer

Figure 3.7: Using the content menus.

Finally, to work faster, you can right-click on the different panels to display a pop-up menu with options related to that particular panel.

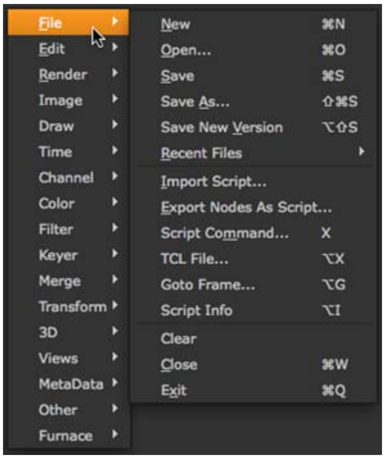

File
Edit
Render
Image
Draw
Time
Channel
Color
Filter
Keyer
Merge
Transform
3D
Views
MetaData
Other
Furnace
New ⌘N
Open... ⌘O
Save ⌘S
Save As... ⇧⌘S
Save New Version ⌥⇧S
Recent Files
Import Script...
Export Nodes As Script...
Script Command... X
TCL File... ⌥X
Goto Frame... ⌥G
Script Info ⌥I
Clear
Close ⌘W
Exit ⌘Q

Figure 3.8: A right-click menu in the Node Graph.

Using the Toolbar

Nuke's Toolbar includes the following icons:

Icon		Functions
	Image	Image read and write nodes, built-in Nuke elements, and Viewer nodes.
	Draw	Roto shapes, paint tools, film grain, fills, lens flares, sparkles, other vector-based image tools.
	Time	Retiming image sequences.
	Channel	Channel management.
	Color	Applying color correction effects.
	Filter	Applying convolve filters, such as blur, sharpen, edge detect, and erode.
	Keyer	Extracting procedural mattes.
	Merge	Layering background and foreground elements.
	Transform	Translating, scaling, tracking, and stabilizing elements.
	3D	3D compositing nodes and tools.
	Views	Nodes for working with views and stereoscopic or multi-view material.
	Metadata	Viewing, editing, and comparing image metadata.
	Other	Additional operators for script and Viewer management.
		Any installed plug-ins and custom menus that do not have their own icon.

To display a tool tip that explains the icon's function, move your mouse pointer over the icon.

To make selections from the Toolbar, click on an icon and select an option from the menu that appears.

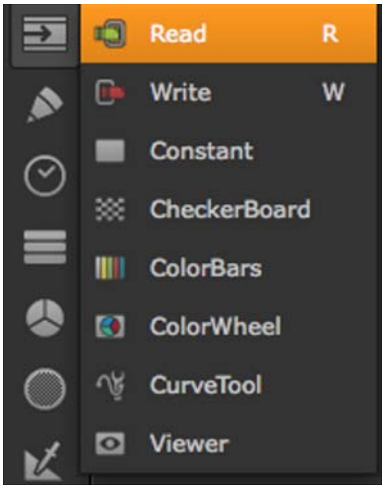


Figure 3.9: Selecting a node from the Toolbar.

To quickly browse through the menus in the Toolbar, click and drag over the icons. Nuke opens and closes the menus as you drag over them, making it easy to search for a particular node or find out what the available menus contain.

Using the Menu Bar

The Nuke menu bar includes these functions:

Menu	Functions
File	Commands for disk operations, including loading, saving, and importing scripts.
Edit	Editing functions, preferences, and project settings.
Layout	Restoring and saving layouts.
Viewer	Adding and connecting Viewers.
Render	Rendering the output.
Help	Accessing a list of hotkeys, user documentation, release notes, training resources, tutorial files, and Nuke-related e-mail lists.

To quickly browse through the available menus and see what they contain, click on a menu and move the mouse pointer over other menus. Nuke opens and closes the menus as you go.

Working with Nodes

Nodes are the basic building blocks of any composite. To create a new compositing script, you insert and connect nodes to form a network of the operations you want to perform to layer and manipulate your images.

Adding Nodes

You add nodes using the Toolbar, the Tab menu, or the right-click menu. When you add a node, Nuke automatically connects it to the last selected node.

To add a node using the Toolbar:

1. To select the existing node that you want the new node to follow, click on the node once.
2. Click an icon on the Toolbar and select a node from the menu that appears. For example, if you want to add a Blur node, click the **Filter** icon and select **Blur**.

Tip *You can press the middle mouse button on a menu icon to repeat the last item used from that menu. For example, if you first select a Blur node from the **Filter** menu, you can then add another Blur node by simply pressing the middle mouse button on the **Filter** icon.*

To add a node using the Tab menu:

1. To select the existing node that you want the new node to follow, click on the node once.
2. Press the **Tab** key and start typing the name of the node you want to create.

   This opens a prompt displaying a list of matches.
3. To select the node you want to add from the list, either:
   - click on it, or
   - scroll to it with the **Up** and **Down** arrow keys and press **Return**.

To add the last node created using this method, simply press **Tab** and then **Return**.

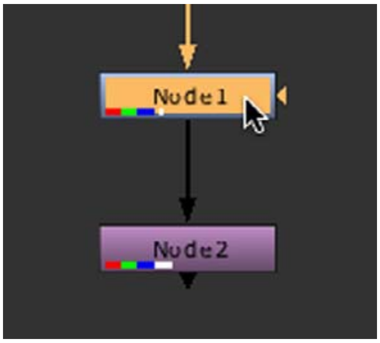
To add a node using the right-click menu:

1. Right-click on the existing node that you want the new node to follow.
2. From the menu that opens, select the node you want to add.

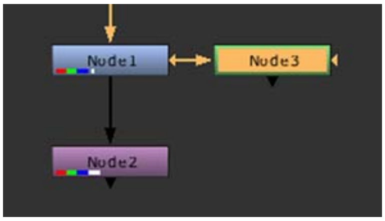
Tip *You can also add nodes using keyboard shortcuts. Most menus in the Toolbar include a note of the relevant hotkey next to the item in question.*

To add a node in a new branch of the node tree:

1. To select the existing node that you want the new node to follow, click on the node once.



2. Hold down **Shift** and create the node using the Toolbar, Tab menu, or right-click menu. To add a node in a new branch with the Tab menu, press the **Tab** key first, then hold down **Shift** when selecting the new node.



The node is added after the selected node in a new branch of the node tree.

Selecting Nodes

Nuke offers a number of options for selecting nodes. Selected nodes display in a highlight color defined in your preferences. The default highlight color is light yellow.

To select a single node:

Click once on the node.

To select multiple nodes:

Press **Shift** while clicking on each node you want to select.
OR
Drag on the workspace to draw a marquee. Nuke selects all nodes inscribed by the marquee.

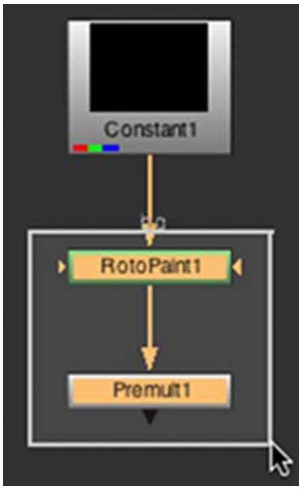


Figure 3.10: Selecting multiple nodes with a marquee.

To select all upstream nodes:

Press **Ctrl** (Mac users press **Cmd**) while dragging a node. Nuke selects all nodes that feed data to the selected node. You can also **Ctrl/Cmd + Shift + click** to select more nodes without clearing the current selection.

To select all nodes in a node tree:

Click on a node in the Node Graph and select **Edit** > **Select Connected Nodes** (or press **Ctrl/Cmd**+**Alt**+**A**).

This selects all nodes in the node tree, whether they are upstream or downstream from the current node. Nodes in any other node trees are not selected.

To select all nodes in a script:

Select **Edit** > **Select all** (or press **Ctrl/Cmd**+**A**).

To select nodes by name:

1. Choose **Edit** > **Search**, or press the forward slash (/).
   A dialog appears.

2. Type an alphanumeric string that is included in the names of the nodes you wish to select. Click **OK**.

Tip *When typing the above alphanumeric search string, you can use asterisks (*) and question marks (?) as wild cards. An asterisk stands for multiple alphanumeric characters. A question mark represents just one character.*

To invert a selection:

Select **Edit** > **Invert Selection**.

Replacing Nodes

To replace one node with another:

1. In the Node Graph, select the node you want to replace.
2. Hold down **Ctrl** (**Cmd** on a Mac) and create a new node using the Toolbar, a right-click menu, or the Tab menu. To replace a node with the Tab menu, press the **Tab** key first, then hold down **Ctrl**/**Cmd** when selecting the new node.

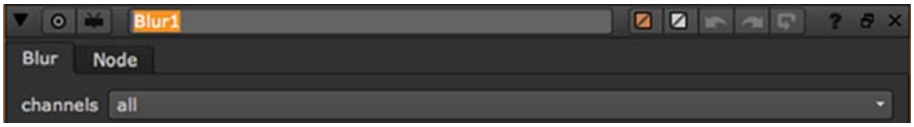
   The new node replaces the selected node in the Node Graph.

Note *Note that you cannot replace nodes in this manner if you are using a hotkey (such as B for the Blur node) to create the new node.*

Renaming Nodes

To rename a node:

1. Double-click on the node to open its properties panel.
2. In the title field on top of the properties panel, you should see the current name of the node. Delete that name and enter a new name in its place,



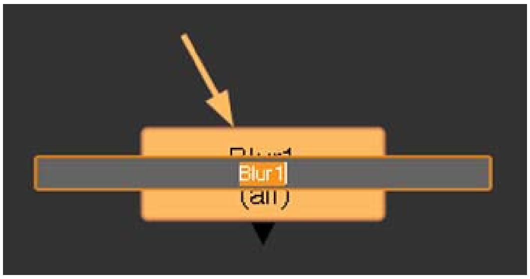
Blur1

channels all

OR

1. Click on the node in the Node Graph to select it.
2. Press N.

3. Enter a new name for the node in the rename field that appears on top of the node.



Blur1

Editing Nodes

To copy, paste, and perform other editing functions in the compositing tree, you can use the standard editing keys (for example, **Ctrl/Cmd+C** to copy, and **Ctrl/Cmd+V** to paste). You can copy nodes to files or memory. Copied nodes inherit the values of their parent, but these values, unlike those in cloned nodes (see below), are not actively linked—that is, you can assign different values to the original and the copy.

When you paste nodes, Nuke automatically connects them to the node that is selected before the paste operation. If you don't want to connect anything, click on a blank area of the workspace to deselect any selected nodes before pasting.

To copy nodes to memory:

1. Select the node or nodes you want to copy.
2. Choose **Edit** > **Copy** (or press **Ctrl/Cmd+C**).

To copy nodes to files:

1. Select the node or nodes you want to copy.
2. Choose **File** > **Export nodes as script**.
3. Navigate to the directory where you want to store the node as a file.
4. Type a name for the node(s) at the end of the pathway, followed by the extension *.nk*.

To cut nodes:

1. Select the node or nodes you want to cut.
2. Choose **Edit** > **Cut** (or press **Ctrl/Cmd+X**).
   Nuke removes the node(s) from the script and writes the node(s) to memory.

To paste nodes from memory:

1. Select the node that you want the pasted node to follow.
2. Choose **Edit** > **Paste** (or press Ctrl/Cmd+V).
   Nuke adds the nodes to the script, connecting them to the node you selected in step 1.

To paste nodes from files:

1. Select the node that you want the pasted node to follow.
2. Choose File > Import script.
3. Navigate to the directory that stores the node file.
4. Select the node file, and click **Import**.
   Nuke adds the nodes described by the file to the node you selected in step 1.

Cloning Nodes

You can clone nodes in preparation for pasting them elsewhere in a script. Cloned nodes inherit the values of their parent, but unlike copied nodes, they also maintain an active link with their parents' values. If you alter the values of one, the other automatically inherits these changes.

Clones are helpful for maintaining consistent setups across multiple elements. For example, you might use clones to apply an identical film grain setup to a series of elements shot on the same stock. Should you need later to make changes to the setup, these changes would automatically ripple throughout the script.

Note *You cannot clone gizmos. This applies to both gizmos created in your facility and the nodes in the Nuke default Toolbar that are in fact gizmos, such as the LightWrap node, the Grain node, and the IBK nodes.*

For more information on gizmos, see "Gizmos, Custom Plug-ins, and Generic TCL Scripts" on page 629.

To clone nodes:

1. Select the node or nodes you want to clone.
2. Choose **Edit** > **Clone** or (press Alt+K).
   Nuke clones the node(s), whilst maintaining an active link to the parental node(s). The clone status is indicated with an orange line that connects the clone to its parent node. The nodes also share the same name.

To declone nodes:

1. Select the node or nodes you want to declone.
2. Choose **Edit** > **Declone** (or press Alt+Shift+K).
   Nuke removes the clone status of the selected nodes.

Disabling and Deleting Nodes

To disable nodes:

1. Select the node or nodes you want to disable.
2. Select **Edit** > **Node** > **Disable\Enable** (or press D).
   Nuke cancels the node(s)'s effect on the data stream.

To re-enable nodes:

1. Select the node or nodes you want to re-enable.
2. Select **Edit** > **Node** > **Disable\Enable** (or press D).

To delete nodes:

1. Select the node or nodes you want to delete.
2. Select Edit > Erase (or press **Delete**).
   Nuke removes the node(s) from the script.

Connecting Nodes

As discussed, when you add or paste nodes into a script, Nuke automatically generates pipes between the currently selected node and the new nodes. As you build up a script, you'll need to move these pipes, or run new pipes between nodes. In Nuke, you make such modifications by dragging on the back end of the pipe (the end without the arrowhead).

To disconnect a node:

Drag the head or tail of the connecting arrow to an empty area of the workspace.
OR
Select the lower node in the tree and press Ctrl+D (Mac users press Cmd+D).

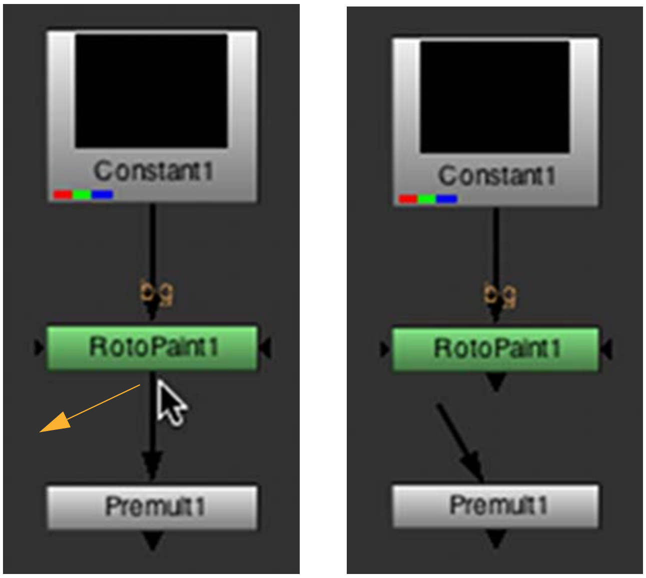

Constant1
bg
RotoPaint1
Premult1
Constant1
bg
RotoPaint1
Premult1

Figure 3.11: Disconnecting a pipe.

To reconnect a node:

Drag on the head or tail of the connecting arrow and drop it over the center of the node to which you want to connect.

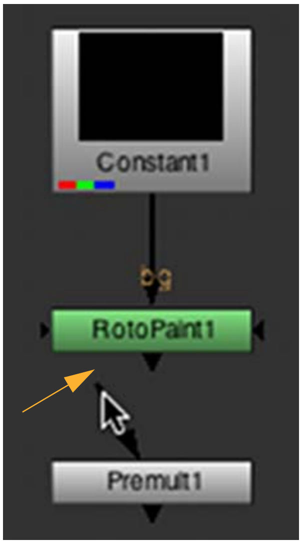
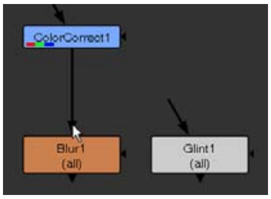

Constant1
bg
RotoPaint1
Premult1

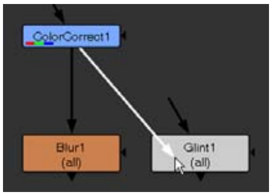
Figure 3.12: Reconnecting a pipe.

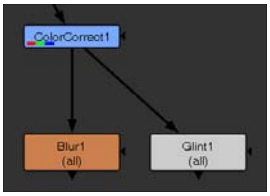
Note *Nuke distinguishes the dual inputs that may run into a merge node with the labels **A** and **B**. **A** refers to the foreground element, and **B** to the background element. Nuke always copies from the **A** input to the **B**. This means that if* *you later decide to disable the node associated with an* ***A*** *input, the data stream will keep flowing, because it will, by default, use the* ***B*** *input.*

To duplicate a connecting arrow:

Shift+drag the connecting arrow on top of the node you want to create a connection to. Nuke duplicates the connecting arrow, leaving the original connection untouched.


ColorCorrect1
Blur1 (all)
Glint1 (all)


ColorCorrect1
Blur1 (all)
Glint1 (all)


ColorCorrect1
Blur1 (all)
Glint1 (all)

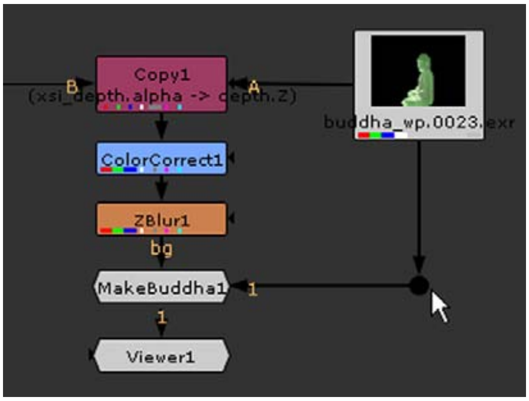
To add a node between two connected nodes:

Drag the node into the space between the already connected nodes. As you do so you will see the link between these two nodes become active. When that happens, simply release the node you are dragging and it will automatically wire itself into the network between the two nodes.

To bend connecting arrows:

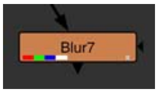
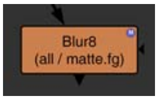
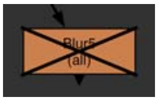
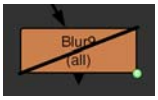
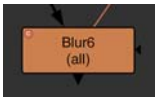
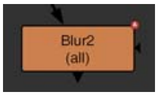
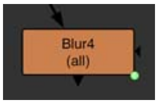
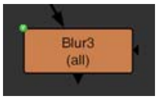
1. Select the node before the connector you want to bend.
2. From the Toolbar, select **Other** > **Dot**. A dot appears after the selected node, causing a bend in the connector.
3. Drag the dot as necessary to reposition the bend.

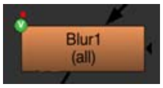
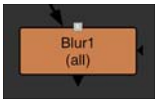
Tip *You can also add a dot to an existing connection by pressing* ***Ctrl (Cmd*** *on a Mac) and clicking on the yellow dot that appears on the connecting arrow.*


Copy1
(xsi_depth.alpha -> depth.Z)
B
A
buddha_wp.0023.exr
ColorCorrect1
ZBlur1
bg
MakeBuddha1
1
1
Viewer1

Indicators on Nodes

There are several indicators that can appear on the nodes in the Node Graph, depending on what you are doing. The following table describes what each indicator means.

Indicator	Where it appears	What it means
		The wide rectangles indicate the channels the node processes. The thin rectangles indicate the channels that are passed through the node untouched.
M		The node's effect is limited by a mask from either the node's primary input or output.
✕		The node has been disabled by pressing **D** or clicking the **Disable** button.
╱		The node has been disabled using an expression.
C		The node has been cloned. The indicator appears on both the parent and the child node.
A		One or more of the node parameters are animated over time.
E		One or more of the node parameters are being driven by an expression.
V		You are working with a multiview project and have split off one or more views in the node's controls.

Indicator	Where it appears	What it means
🟥		You are working with a multiview project and have split off one or more views in the node's controls, dots also appear on the node to indicate which views have been split off. For example, if you are using red for the left view and split off that view, a red dot appears on the node.
X		The full effect of the node is not in use, because you have adjusted the **mix** slider in the node's controls.

Searching for Nodes

Nuke's Search feature allows you to search for nodes in your script and select any matches found. As a search string, you can enter all or part of a node name. For example, you can search for all Blur nodes in your script by entering **bl** as the search string.

Using regular expressions, you can also do more complex searches, such as searching for all the Read and Write nodes in a script.

To search for nodes:

1. Select **Edit** > **Search** (or press /) to bring up the search dialog.
2. In the search field, enter the string you want to search for.

   If you want to search for all nodes in the script, enter * (an asterisk).

   If you want to search for all Read nodes in the script, enter **Read**.

   If you want to search for all the Read and Write nodes, enter the following expression:

   (*Read*|*Write*)
3. Click **OK**.

Nuke searches for the nodes in the script and selects any matches it finds.

Note *When you enter expressions in the search dialog, remember that the search field only takes regular expressions. Any characters that have specific meanings in regular expressions, such as [and], need to be preceded by the \ character.*

Viewing Information on Nodes

You can obtain more detailed information from any node by selecting that node and then pressing the 'i' key. This will display an information window associated with that node, particularly useful when troubleshooting.

Customizing the Node Display

You can modify the color, name, and notes that a particular node displays. Doing so can make it easier for other artists to decipher the intent of your script. For example, you might color all nodes green that relate to keying.

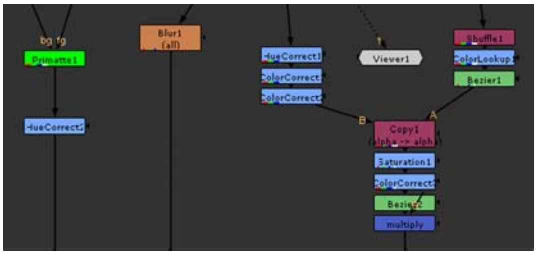

bg fg
Primatte1
HueCorrect2
Blur1 (all)
HueCorrect1
ColorCorrect
ColorCorrect
Viewer1
Shuffle1
ColorLookup1
Bezier1
B
A
Copy1 (alpha -> alph
Saturation1
ColorCorrect
Bezier2
multiply

Figure 3.13: Nodes can be color-coded according to function.

To modify a node's display characteristics:

1. Double-click on the node to display its properties panel.
2. Click the **Node** tab at the top of the dialog. Its attributes appear.
3. Do any of the following:
   - Enter a new name for the node on top of the old name.
   - Click on the left color button to change the color of the node. To copy a color of one node to another, drag and drop the color button from the node whose color you want to copy on top of the color button in the node whose color you want to change.
   - Type any comments regarding the node in the **label** field. These will appear on the surface of the node.
   - From the **font** pulldown menu, select the font type for any text on the node.
   - Use the buttons on the right to bold or emphasize the text.
   - Enter the font size in the font size field.
   - Click **color** to choose a new font color.
   - The *Select color* dialog appears, allowing you to select the desired color.

Tip *You can also have Nuke automatically color-code nodes for you based on their function. Select **Edit** > **Preferences** (or press **Shift+S**) to open the Preferences dialog. Click on the **Node Colors** tab, and check the **autocolor** option.*

*Thereafter, every time you add a node to a script, the node will take on the color appropriate to its function. You can edit these colors as necessary on the **Node Colors** tab of the Preferences dialog.*

- Check **hide input** to conceal the node's incoming pipe. This can enhance the readability of large scripts.
- Check **postage stamp** to display a thumbnail render of the node's output on its surface.

Navigating Inside the Node Graph

As scripts grow in complexity, you need to be able to pan to a particular cluster of nodes quickly. The Node Graph offers a couple of methods for doing so.

Panning

Panning with the mouse

To pan with the mouse, press the middle mouse button and drag the mouse pointer over the workspace (you can also use Alt+drag). The script moves with your pointer.

Note *On Linux, **Alt**+drag may not work as expected. This is due to the default window functionality on Gnome. To get around it, you can use the **Windows** key instead of **Alt** when panning.*

Alternatively, you can change your window preferences on Gnome to fix the problem:

*1. Select **Applications** > **Preferences** > **Windows** to open the Window Preferences dialog.*

*2. Under **Movement Key**, select **Super ("or Windows logo")**.*

*You should now be able to pan with **Alt**+drag.*

Panning with the map

If your script spills over the edges of the workspace, a navigator map automatically appears in the bottom-right corner.

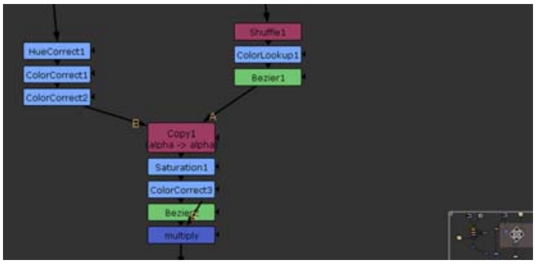

HueCorrect1
ColorCorrect1
ColorCorrect2
Shuffle1
ColorLookup1
Bezier1
B
A
Copy1
alpha -> alpha
Saturation1
ColorCorrect3
Bezier2
multiply

Figure 3.14: Panning with the map.

The map shows you a miniature view of the entire script and the pink rectangle shows the portion of the script that you see within the workspace borders.

To pan with the map, drag the pink rectangle to pan to a different view of the script.

When you pan or resize the window, the map automatically appears when the workspace does not show the entire script. When the whole script is contained within the window border, then the map automatically disappears.

Tip *The navigation map is resizeable. Drag on its upper left corner to make it as large or small as you like.*

Zooming

You can zoom in on or out from the script in a couple of ways.

To zoom in:

Move your mouse pointer over the area you want to zoom in on, and press the plus key (+) repeatedly until the workspace displays the script at the desired scale.
OR
Press **Alt** and drag right while holding down the middle mouse button.

To zoom out:

Move your mouse pointer over the area you want to zoom out from, and press the minus key (-) repeatedly until the workspace displays the script at the desired scale.
OR
Press **Alt** and drag left while holding down the middle mouse button.

Note *On Linux,* ***Alt****+middle drag may zoom the entire Nuke window instead of the Node Graph. This is default functionality on Gnome. To get around it, you can use the* ***Windows*** *key instead of* ***Alt*** *when zooming.*

Alternatively, you can change your window preferences on Gnome to fix the problem:

1. Select ***Applications > Preferences > Windows*** *to open the Window Preferences dialog.*

2. Under ***Movement Key****, select* ***Super ("or Windows logo")****.*

You should now be able to zoom in and out of the Node Graph with ***Alt****+middle drag.*

Fitting Selected Nodes in the Node Graph

To fit selected nodes in the Node Graph, click the middle mouse button or press F.

Fitting the Node Tree in the Node Graph

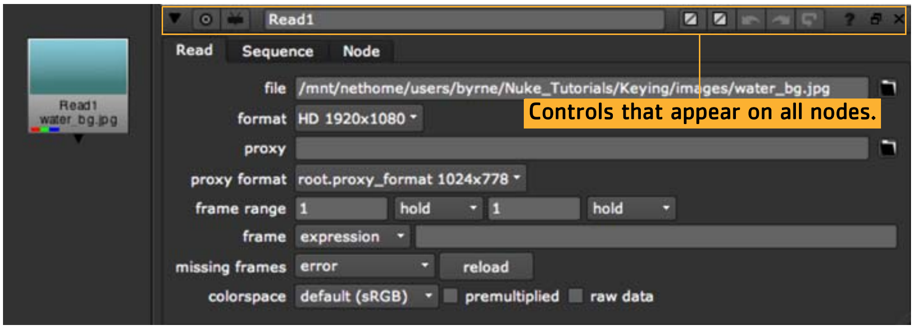
To fit the entire node tree in the Node Graph, click on the Node Graph to make sure no nodes are selected and click the middle mouse button or press F.

Properties Panels

When you insert a node, its properties panel appears in the Properties Bin with options to define the node's output. You can also open the properties panel later by doing any of the following:

- Double-click on the node in the Node Graph.
- Ctrl+click on the node in the Node Graph (Mac users Cmd+click).
- Select the node in the Node Graph and press **Return**.

Tip *To open a properties panel in a floating window,* ***Ctrl+Alt****+click (Mac users* ***Cmd+Alt****+click) on the node.*



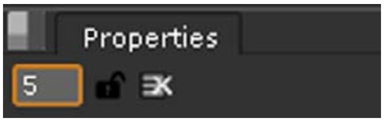
Controls that appear on all nodes.

file /mnt/nethome/users/byrne/Nuke_Tutorials/Keying/images/water_bg.jpg
format HD 1920x1080
proxy
proxy format root.proxy_format 1024x778
frame range 1 hold 1 hold
frame expression
missing frames error reload
colorspace default (sRGB) ☐ premultiplied ☐ raw data

Figure 3.15: The Read node and its properties panel.

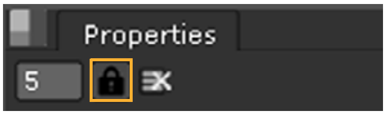
Managing the Properties Bin

You can limit the number of properties panels that can be open in the Properties Bin. To do so, enter the maximum number of properties panels in the field on the Properties Bin.

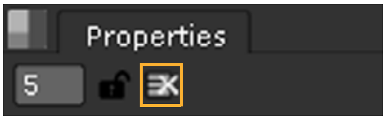


5

To lock the Properties Bin and have all new properties panels appear in floating windows, click the lock button on the Properties Bin.



To empty the Properties Bin and close all the properties panels in it, click the remove all panels button.



Tip *You can also close all the properties panels in the Properties Bin by* ***Alt****+clicking on the close (****X****) button of one of the panels.*

Controls That Appear on All Properties Panels

These are the standard controls of every properties panel:

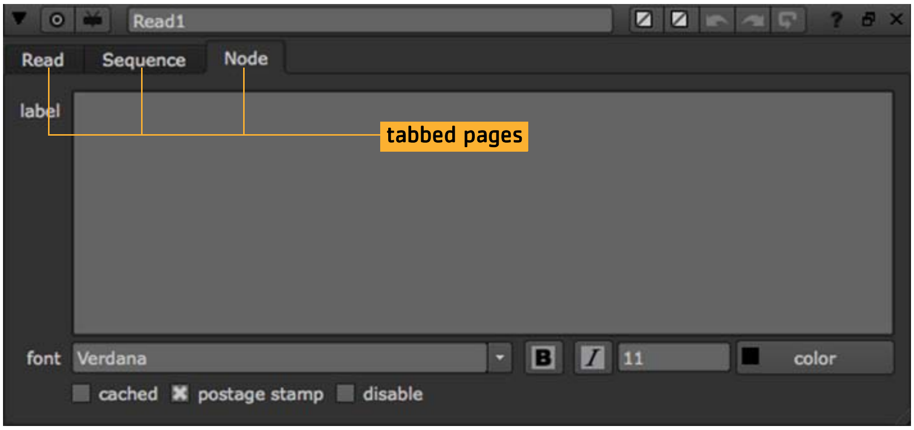
Control	Function
▼ ▶	Hide or show the node's tabbed pages.
◎	Centers the node in the Node Graph.
✕	Centers one of the node's inputs in the Node Graph. Select the input from the pulldown menu that appears.
name field (for example, **Blur1**)	You can enter a new name for the node here.
◪ (left)	Changes the color of the node. You can drag and drop this button on top of another color button to copy the color. To revert to the default color defined in your Preferences, right-click on the button and select **Set color to default**. An X on the button indicates the color is unset, and the color defined in the Preferences is used.
◪ (right)	Changes the color used for the node's controls in the Viewer. You can drag and drop this button on top of another color button to copy the color. To revert to the default color defined in your Preferences, right-click on the button and select **Set color to default**. An X on the button indicates the color is unset, and the color defined in the Preferences is used.
↶	Undoes the last change made to the node.

Control	Function
↷	Redoes the last change undone.
↺	Reverts any changes made after the properties panel was opened.
?	Displays a pop-up help related to the node and its controls.
⧉	Floats the properties panel. Clicking this button again docks the properties panel back in the Properties Bin if the Bin exists.
✕	Closes the properties panel. **Alt**+click this to close all the properties panels in the Properties Bin. **Ctrl/Cmd**+click to close all properties panels except the one clicked on.

Floating control panels also include the following buttons:

Control	Function
Revert	Reverts any changes made after the properties panel was opened.
Cancel	Reverts any changes made after the properties panel was opened and closes the properties panel. Hitting this button right after a node was created also deletes the node from the Node Graph.
Close	Closes the properties panel.

Many properties panels also contain several tabbed pages.



label

font Verdana 11 color

☐ cached ☒ postage stamp ☐ disable

Figure 3.16: The Read node's properties panel and the Node tab.

On the **Node** tab, you can usually adjust the following controls:

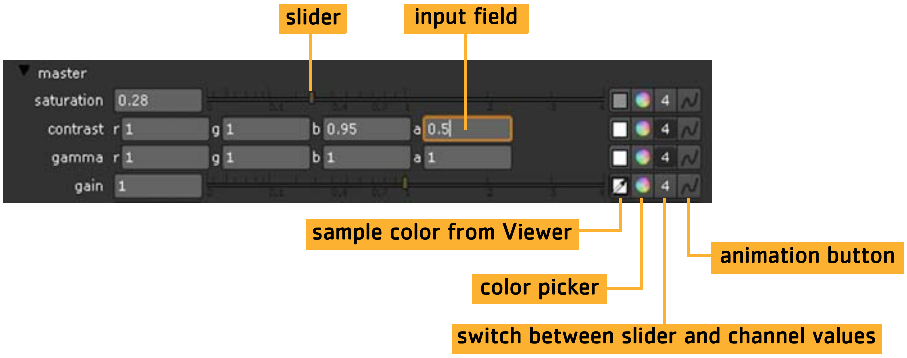
Control	Function
label	Lets you add comments to the node. The comments are displayed on the node's surface. If you like, you can use HTML in the label field. For example, to have your comments appear in bold, you can enter My Comment. To add an icon called *MyIcon.png* to the node, you can use . Save the icon in your plug-in path directory. (For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.) Most common image formats will work, but we recommend using .png. Note that the HTML has been changed to a slightly non-standard form where newlines are significant. If there is a newline character in your data, a new line will be displayed in the label.
font	Lets you change the font for any text displayed on the node.
B	Bolds any text displayed on the node.
I	Emphasizes any text displayed on the node.
11	Lets you change the font size of any text displayed on the node.
🟦 color	Lets you change the color of any text displayed on the node.
hide input	Check this to hide the node's incoming pipe. This control does not appear on all nodes.
cached	Check this to keep the data upstream from the node in memory, so that it can be read quickly. When this is checked, a yellow line displays under the node in the Node Graph.
postage stamp	Check this to display a thumbnail render of the node's output on its surface.
disable	Check this to disable the node. Uncheck to re-enable. (You can also disable or re-enable a node by selecting it in the Node Graph and pressing D.)

Displaying Parameters

To display a node's parameters:

Double-click the node. It's properties panel appears.

Figure 3.17 shows the controls available for editing parameters. Note that the presence of each control will vary according to the parameter's function.



master
saturation 0.28
contrast r 1 g 1 b 0.95 a 0.5
gamma r 1 g 1 b 1 a 1
gain 1

Figure 3.17: Parameter controls.

Using Input Fields

You can key values directly into a field, or use the arrow keys to increment and decrement values.

To key in field values:

1. Click in the field to select the value you wish to replace:
   - Double-click to the left of the value to select only the whole number digits.
   - Double-click the right of the value to select only decimal digits.
   - Press **Return** or double-click on the decimal itself to select all digits.
2. Type the value you want to replace the selection.

Tip *You can also enter expressions (programmatic instructions for generating values) into fields. You always start an expression by typing =. See Chapter 18, "Expressions", on page 533 for information about how to format expressions.*

Tip *Nuke also allows you to enter formulas into fields, making it easy to do quick calculations. For example, if you wanted to halve a value of 378, you could simply type **378/2** into a field and press **Enter** to get 189.*

You can increment or decrement field values by hundreds, tens, tenths, hundredths, and so on. The magnitude of change depends on the initial position of your cursor. For example if you wanted to increment the initial value of **20.51** by ones, you would insert your cursor before the **0**.

To increment and decrement a field value:

1. Click to insert the cursor just prior to the digit you want to increment or decrement.

2. Press the **up arrow** to increment by unit, or the **down arrow** to decrement by one unit.

Tip *You can also increment and decrement values using the mouse wheel (if available) or by pressing **Alt** while dragging on the value. The latter method is particularly useful for tablet users.*

Using Sliders

To set a value with a slider:

Drag the knob to the desired value.
OR
Click the desired value on the graduated scale beneath the slider.

To reset a slider to its default value:

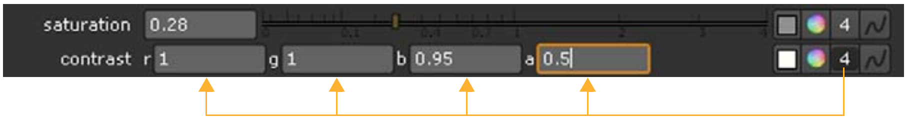
Ctrl+click (Mac users **Cmd**+click) on the slider.

Separating Channels

By default, many parameters in Nuke automatically gang channels for you. For example, if you drag on the Gain slider in the ColorCorrect node, you simultaneously affect the R, G, and B channels (assuming you're processing the RGB channel set). You can, however, use the parameter's channel chooser button to reveal and edit the values assigned to individual channels.

To edit an individual channel's value:

1. Click the parameter's channel chooser button. The number on its surface lets you know how many channels are available for editing. A series of input fields—one for each individual channel—appears.



saturation 0.28
contrast r 1 g 1 b 0.95 a 0.5

2. Edit the values in any of the revealed input fields as necessary.

You can also use a color slider for editing individual channels. See "Using Color Sliders and Color Wheel" on page 67.

Using the Color Picker and Color Controls

Nuke offers a color picker for inputting values. Color pickers are especially useful for setting white or black points, or for color matching tasks, such as sampling values in a background plate to color-correct a foreground image.

To use the color picker and color controls:

- Click the color picker button to display and make color selections. The color picker window opens.
- To activate the eye dropper and sample colors from the Viewer (see below), click the color swatch button.

  To copy a color from one color swatch to another, drag and drop the color swatch with the desired color on top of the swatch whose color you want to change.
- To toggle between using the slider and manually entering values for each of the channels, click the channel chooser button (which displays the number of available channels).

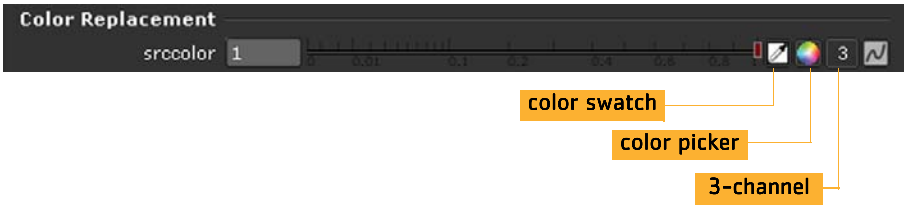


Figure 3.18: Options for color selection.

To sample a color from the Viewer:

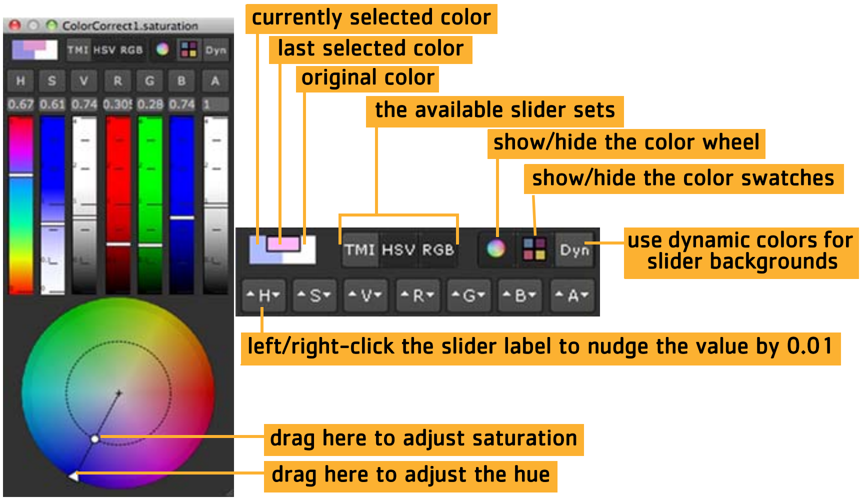
1. In the parameters, click the color swatch to activate the eye dropper.
2. Move your mouse pointer over the Viewer from which you wish to copy color values.
3. Zoom and pan as necessary until the region from which you want to sample is clearly visible.
4. **Ctrl/Cmd**+click to sample a color value from the Viewer, or **Ctrl/Cmd+Alt**+click to sample a color from the node's input while viewing its output. To sample a region rather than a single pixel's color value, also press **Shift**. The input fields associated with the color picker update to reflect the color values of the sampled pixels. In the case of a sampled region, Nuke inserts the average color values of all inlying pixels.
5. If you didn't manage to sample just the right pixel or quite the right region, **Ctrl/Cmd**+click, **Ctrl+Alt**+click, **Ctrl/Cmd+Shift**+click, or **Ctrl/Cmd+Alt+Shift**+click again. A new overlay appears and the old one disappears.
6. When you've captured the right values, click the color picker button again. The color swatch reappears displaying the color or color average you sampled.

Using Color Sliders and Color Wheel

Some parameters offer color sliders and a color wheel for inputting values. These offer an intuitive and precise means for assigning just the right color value to a parameter.

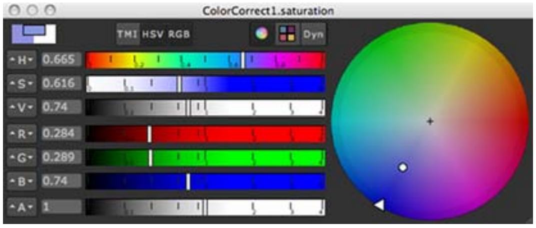
To display the color sliders and color wheel:

Click the parameter's color picker button. The color sliders and wheel appear.


ColorCorrect1.saturation
TMI HSV RGB Dyn
H S V R G B A
0.67 0.61 0.74 0.305 0.28 0.74 1
currently selected color
last selected color
original color
the available slider sets
show/hide the color wheel
show/hide the color swatches
use dynamic colors for slider backgrounds
left/right-click the slider label to nudge the value by 0.01
drag here to adjust saturation
drag here to adjust the hue

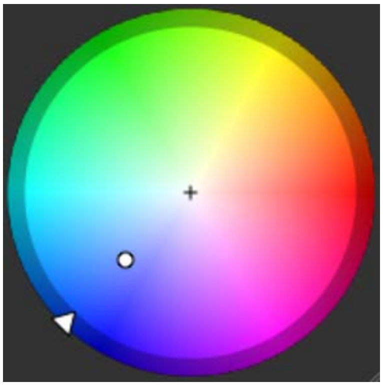
To customize the color sliders and wheel window:

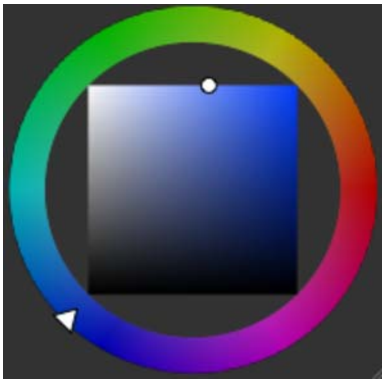
- From the **TMI**, **HSV**, and **RGB** buttons, select which slider set you want to display. The available slider sets are described below.
- To make the sliders horizontal rather than vertical, resize the color sliders window. When the window is wide enough, the sliders become horizontal.


ColorCorrect1.saturation
TMI HSV RGB Dyn
H 0.665
S 0.616
V 0.74
R 0.284
G 0.289
B 0.74
A 1

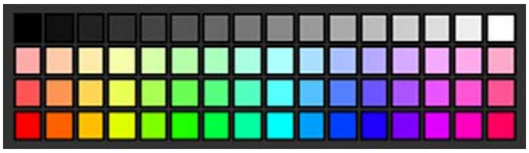
- If you want the background of the sliders show what the value of the color would be if the sliders were set to the current position, click the **Dyn** button.

- Click the color wheel button to cycle through three states: the color wheel, the color square, and hide color wheel/square.



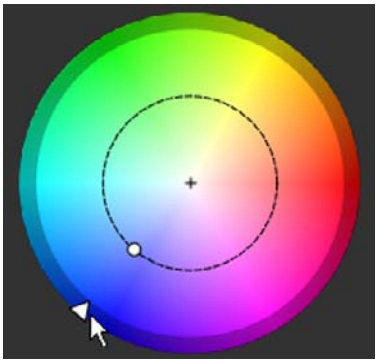


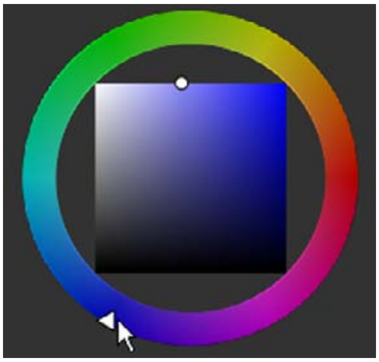
- To hide or show the color swatches, toggle the color swatches button.



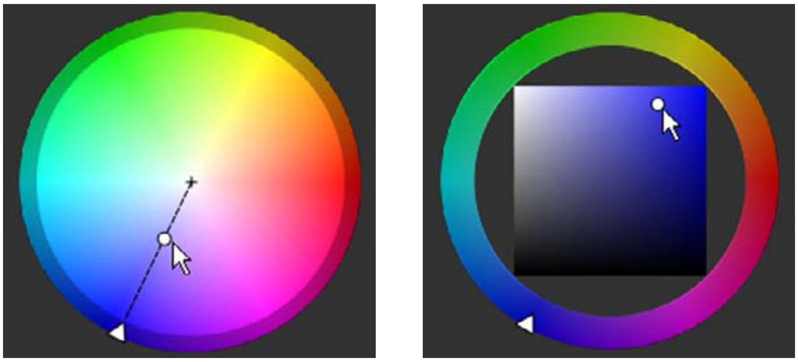
To use the color wheel:

- To adjust the hue, drag the marker on the edge of the color wheel/on the circle.





- To adjust the saturation, drag the marker inside the wheel/square.



Tip *You can also **Ctrl/Cmd**+click on the color wheel to only affect the hue of the selected color, and **Shift**+click to only affect the saturation.*

- To zoom in and out of the color wheel, press **Alt** and drag right or left with the middle mouse button.
- To pan in the color wheel, press **Alt** and drag the mouse pointer over the color wheel.
- To reset the zoom and/or pan, middle-click on the color wheel.

To use the color sliders:

To increment the value by 0.01, right-click on the slider label (e.g. **R** or **A**). To decrement the value by 0.01, left-click on the label. Use **Shift**+click for 0.1, and **Alt**+click for 0.001.

You can also click and drag right or left on a label to scrub the value up or down. Use **Shift**+drag to scrub faster, and **Alt**+drag to scrub slower.

These are the functions of the TMI sliders:

- **The temperature slider (T)** lets you control apparent color temperature by inversely affecting red and blue values (assuming your are processing the RGBA channel set).

  To cool (that is, increase the blue channel's value, while decreasing the red channel's), drag up. To heat (increase the red channel's value, while decreasing the blue channel's), drag down.
- **The magenta/green slider (M)** lets you control the mix of green and magenta hues.

  To add more magenta (increase the red and blue channel's values, while decreasing the green channel's), drag up. To add more green (increase the green channel's value while decreasing the red and blue channels'), drag down.

- **The intensity slider (I)** lets you simultaneously control the red, green, and blue channel values.

  To increase the value of all channels by the same amount, drag up. To decrease the value of all channels by the same amount, drag down.

  To increase the channel's value, drag up. To decrease it, drag down.

These are the functions of the HSV sliders:

- **The hue slider (H)** lets you control the color's location on the traditional color wheel (for example, whether the color is red, yellow, or violet).
- **The saturation slider (S)** lets you control the intensity or purity of the color.
- **The value slider (V)** lets you control the brightness of the color (the maximum of red, green, and blue values).

These are the functions of the RGB sliders:

- **The red slider (R)** lets you control the red channel's value (or the first channel in a channel set if you are processing another set besides RBGA).

  To increase the channel's value, drag up. To decrease it, drag down.
- **The green slider (G)** lets you control the green channel's value (or the second channel in a channel set if you are processing another set besides RBGA).

  To increase the channel's value, drag up. To decrease it, drag down.
- **The blue slider (B)** lets you control the blue channel's value (or the third channel in a channel set if you are processing another set besides RBGA).

  To increase the channel's value, drag up. To decrease it, drag down.

The alpha slider is included in all three slider sets:

- **The alpha slider (A)** lets you control the alpha channel's value (or the fourth channel in a channel set if you are processing another set besides RBGA).

  To increase the channel's value, drag up. To decrease it, drag down.

To use the color swatches:

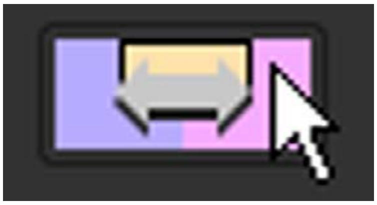
When you have found a good color, you may want to save it in one of the color swatches for further use. To do so, adjust the color wheel or sliders until you are happy with the color, and right-click on the color swatch where you want to save it. You can also drag and drop a color into a color swatch from any other color button or swatch.

To open another color picker window:

To open another color picker window while keeping the first window open, **Ctrl/Cmd**+click on another parameter's color picker button.

To switch between the current and previous or original color:

The rectangle above the sliders shows the original color (on the right) next to the currently selected color (on the left). When you drag the markers to adjust the color, the last selected color is shown in between these. To switch between the currently selected color and the original color, click on the rectangle.



Animating Parameters

Animating a parameter refers to changing its value over time. You do so by setting key frames (frames at which you explicitly define a value) and allowing Nuke to interpolate the values in between. You can animate most of Nuke's parameters in this manner.

Working with Animated Parameters

The Animation menu lets you set key frames, delete keys, and perform other editing operations on the curves for animated parameters.

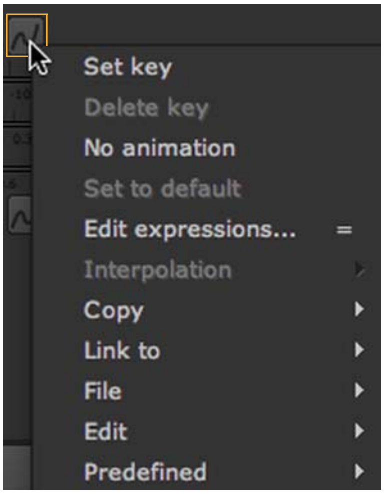


Figure 3.19: The animation menu.

To set key frames:

1. Use a Viewer to cue to a frame where you want to place a key.
2. Click the animation button next to the parameter you want to animate.
3. Select **Set key** from the drop down menu. The parameter's input field turns blue, indicating that a keyframe has been inserted. Nuke enters the autokey mode: when you change the parameters value at another frame, it will automatically create a keyframe for you.

   You can also set a key for all the controls in a node. To do so, select **Set key on all knobs** from the properties panel right-click menu.
4. Cue to the next frame where you want to place a key.
5. Edit the parameter's value using the input field, regular slider, or color slider. The moment you change the value, Nuke creates a keyframe.
6. Continue adding key frames as necessary.
7. Use the Viewer's scrubber to preview the result.

To delete a single keyframe:

1. Use the Viewer's next keyframe and previous keyframe buttons to cue to the keyframe that you want to remove. Notice that the scrub bar indicates key frames with a blue mark.
2. Click the animation button.
3. Select **Delete key** from the drop down menu.

Nuke removes the keyframe.

To delete all key frames from a parameter:

1. Click the animation button.
2. Select **No animation** from the drop down menu. A confirmation dialog appears. Select **Yes**.

   Nuke removes all key frames from the parameter, and sets the static value to match that of the current frame.

To link animated parameters with a Tracker node

You can link any controls with others by using expressions (see "Expressions" on page 533), but linking with the Tracker node is even easier than that if you use the **Link to** option in the Animation menu. For example to link the **translate** control of the Roto node with a Tracker node, do the following:

1. Create the Tracker node you want to link to.
2. Click on the translate animation menu in the Roto control panel.
3. Select **Link to** > **Tracker linking dialog**
4. Select the Tracker node you want to use in the tracker node dropdown and in the **link to** dropdown, select whether you want to link to the position of the track or the translate values of it.
5. Select which tracks you want to use by checking the **track** boxes. The Expression field updates with the appropriate expression syntax. Then click **OK**, and your linking is done.

   Your Bezier shape's translate value now changes when the Tracker value is changed.

Animated Parameters in the Dope Sheet and the Curve Editor

As you add key frames to a parameter, Nuke automatically both adds markers to the Dope Sheet for them and plots a curve on its Curve Editor panel, where each value (the y axis) is plotted as it changes over time (the x axis). You can add key frames, delete key frames, and even adjust the interpolation between key frames without ever looking at this curve. However, as the animation grows more complex, you may find it easier to edit the animation by manipulating this curve and the keyframes directly. For more information on how to do so, see "Working with Keyframes in the Dope Sheet" on page 74 and "Using the Curve Editor" on page 77.

Working with Keyframes in the Dope Sheet

The Dope Sheet gives you access to all the keyframes you've created, and provides an easy way of editing them. You can also use it to reposition Read nodes and trim them.

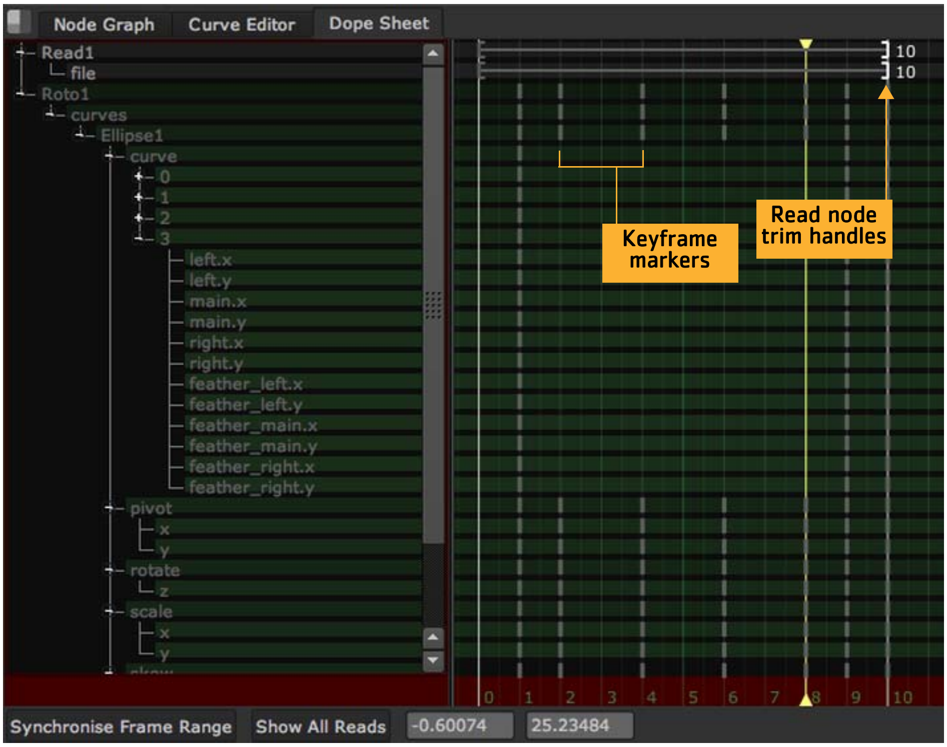
Viewing Keyframes in the Dope Sheet

To use the Dope Sheet for editing your script, you need to create some Read nodes or keyframes first. The Dope Sheet displays all your Read nodes that have their control panels open, and any nodes you've used to create keyframes. To open the Dope Sheet do the following:

1. Click the **Dope Sheet** tab to open it and view your Read nodes and keyframes. If you can't see the tab, right-click the Node Graph title bar and select **Dope Sheet**.
2. Click through the hierarchy on the left to see your nodes and a hierarchical list of animated controls.

   Your keyframes display on the right view as gray lines in the keyframe grid. If you go down to the lowest level of hierarchy, you can also view the keyframes set for different views in stereoscopic projects.

   You can also open a node's control panel by clicking the node in the hierarchy view.


Node Graph
Curve Editor
Dope Sheet
Read1
file
Roto1
curves
Ellipse1
curve
0
1
2
3
left.x
left.y
main.x
main.y
right.x
right.y
feather_left.x
feather_left.y
feather_main.x
feather_main.y
feather_right.x
feather_right.y
pivot
x
y
rotate
z
scale
x
y
10
10
Keyframe markers
Read node trim handles
0 1 2 3 4 5 6 7 8 9 10
Synchronise Frame Range
Show All Reads
-0.60074
25.23484

Synchronising the Frame Range Between the Dope Sheet and the Curve Editor

You can synchronize the frame range of your project between the Dope Sheet and the Curve Editor. This is useful when you're using both of them to edit your animation curves and keyframes at the same time. To synchronize the frame range:

1. Right-click anywhere on the Dope Sheet.
2. Select **Edit** > **Synchronise frame range**. Any changes you make in the Dope Sheet are now applied in the Curve Editor too, and vice versa. Zooming in and out in either also zooms the other view.

   You can also do this by clicking the **Synchronise Frame Range** button at the bottom of the Dope Sheet.

Selecting and Editing Keyframes

To select keyframes

- To select multiple keyframes, drag a marquee over your keyframes in the keyframe view.
- To select a single keyframe, click it in the keyframe view. Its parent keyframe and all other keyframes depending on it are selected too. The frame number of the frame displays next to the frame marker line.
- To select all keyframes in one node, click the node name in the hierarchy view. All the child keyframes under that node are selected.

To move and rescale keyframes

After you've selected keyframes, you can move and rescle them:

- To move your keyframes, drag the selection around in the keyframe view.
- To nudge the selected keyframes right or left, use the keypad arrow keys.
- To edit the frame number of your keyframe directly, double-click on the frame number next to the keyframe and adjust it in the **x** field box that pops up.
- You can also move your keyframes with the move control at the bottom of the Dope Sheet: enter the number of frames by which you want to move the selected keyframes, and click **Move**. You can enter negative values to move the keyframes backwards.
- To rescale, grab the right or the left side of your marquee selection box and drag it. The opposite side of the box stays put and the keyframes are proportionately rescaled according to how you adjust the box.

To delete and add keyframes in the Dope Sheet

You can remove keyframes in the Dope Sheet. To delete a keyframe, just select it and press **Delete/Backspace** button or right-click on the keyframe and select **Edit** > **Erase**.

You can also add keyframes in the Dope Sheet to add animation where there isn't any at the moment. Add keyframes by **Ctrl/Cmd+Alt**+clicking on an empty spot in the Dope sheet. A new keyframe is created and you can adjust it as necessary.

Repositioning and Trimming Read Nodes

You can reposition and trim your Read nodes in the Dope Sheet using the Read node handles.

To view all your Read nodes in the Dope Sheet:

By default, only the Read nodes that have control panels open show up on the Dope Sheet. If you click the **Show All Reads** button, all Read nodes on your script are added in the hierarchy view. If you click the button again, the default view is restored.

You can also open a node's control panel by clicking the node in the hierarchy view.

To position your Read node:

1. Select the Read node you want to move in the hierarchy view. The lines signifying the frame range of that Read node are selected.
2. Drag the Read node lines right or left depending on where you want to position it.

To trim a Read node:

1. Select the Read node you want to move in the hierarchy view. The lines signifying the frame range of that Read node are selected.
2. Move the Read node handles in the keyframe view to adjust the frame range of your Read node. The **frame range** control in the Read node control panel is adjusted automatically.
3. If you want, you can change how the sequence displays outside of the frame range you've defined. Adjust the **before** and **after** dropdown lists next to the **frame range** control in the Read node control panel to change what happens before and after the frame range limits correspondingly:
   - **hold** - select to show a still picture of the first/last frame of the frame range.
   - **loop** - select to start over and keep looping the span of the frame range outside the first/last frame of the frame range.

- **bounce** - select to play the span of the frame range backwards and forwards between the frame range limits.
- **black** - select to display a black frame outside of the first/last frame.

Using the Curve Editor

The Curve Editor enables you to edit curves without physically entering information in the Properties Bin.

Displaying Curves

To reveal an animation curve:

1. Click the animation button next to the parameter whose curve you wish to view.
2. Select **Curve Editor**. The Curve Editor panel appears with a focus on the selected parameter's curve.

   The vertical, or y axis, denotes the value of the parameter.

   The horizontal, or x axis, denotes time (in frame units).

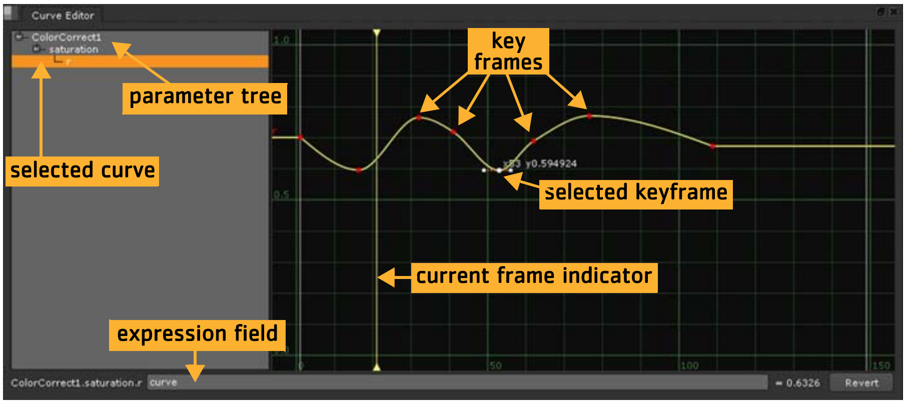

Curve Editor
ColorCorrect1
saturation
r
parameter tree
selected curve
key frames
x53 y0.594924
selected keyframe
current frame indicator
expression field
1.0
0.5
0
50
100
150
ColorCorrect1.saturation.r
curve
= 0.6326
Revert

Figure 3.20: Nuke's Curve Editor panel.

To display curves in the Editor:

1. In the parameter tree on the left, click the + and – signs to expand and collapse the hierarchy as necessary.
2. Click a parameter's name to make its curve the focus of the editor. To focus on multiple curves at the same time, **Shift**+click on the names in the parameter tree.
3. To display separate curves for each channel, separate the channels for the relevant control in the node's properties panel.

The parameter tree on the left lets you focus on any curve in the script.

To remove a curve from the Editor:

1. In the parameter tree on the left, click the + and - signs to expand and collapse the hierarchy as necessary.
2. Select a curve in the parameter tree, and press **Delete**.

To zoom in or out in the Editor:

1. Click on the area you want to zoom in on or out of.
2. Press the + button to zoom in, or the - button to zoom out,

   OR

   Scroll up with the mouse wheel to zoom in, or down to zoom out.

Tip *To zoom to a custom area in the Curve Editor, middle-click on the Editor and drag to select an area with a marquee. When you release the mouse button, the Editor will zoom to fit the selected area in the Editor.*

To pan in the Editor:

Hold the middle mouse button and drag over the Editor. You can also use **Alt**+drag.

To reset zoom and panning:

1. Right-click on the Curve Editor.
2. From the menu that opens, select **View** > **Frame All**.

   OR

   Press **A** on the Editor.

   OR

   Click the middle mouse button.

   Nuke centers the curve in the Editor, resetting the zoom.

To center a portion of the curve in the editor:

1. Select the points you want to center in the editor.
2. Right-click on the Editor, and select **View** > **Frame Selected** (or press **F** on the Editor).

   Nuke centers the selected portion of the curve in the editor. If no points are selected, Nuke centers the selected curve, or all curves.

Editing Curves

You edit curves by moving the points on the curve to new locations. If necessary, you can add more points to the curve. You can also sketch curves freely, use the usual editing functions such as copy and paste, smooth curves with filtering, interpolate curves, loop, reverse or negate curves, and use expressions to modify curves.

To add points to a curve:

1. Click on the curve you want to edit. The curve turns yellow to indicate it's selected.
2. **Ctrl**+**Alt**+click (Mac users **Cmd**+**Alt**+click) on the part of the Curve Editor you want to add a point to. You can add points both on the curve and outside the curve,

   OR

1. Right-click on the Editor and select **Edit** > **Generate**. The *Generate keys* dialog opens.
2. In the **Start at** field, enter the first frame you want to use as a keyframe.
3. In the **End at** field, enter the last frame you want to use as a keyframe.
4. In the **Increment** field, enter the frame increment you want to use between the first and the last keyframe. For example, if you want every tenth frame to be a keyframe, enter **10**.
5. In the last field, enter the value you want to use for y. If you do not enter a value here, the key frames are added to the current curve without modifying the curve shape.
6. Click **OK**.

To select points on a curve:

- To select individual points, click on the point you want to select.
- To select multiple points, **Shift**+click on the points, or drag a marquee around them.

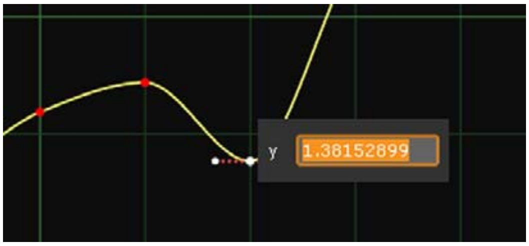
  A box is drawn around the points, and the points turn white to indicate they have been selected.
- To select all points, press **Ctrl**+**A** (Mac users press **Cmd**+**A**).

  A box is drawn around the points, and the points turn white to indicate they have been selected.

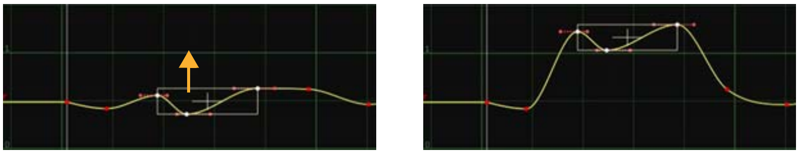
To move points on a curve:

- To move a point along either the x or y axis only, drag the point to a new location.
- To move a point in any direction, **Ctrl**+drag (Mac users **Cmd**+drag) the point to a new location. You can also nudge points using the numeric keypad arrows.

- To adjust the values of a point numerically, select the point and double-click on the x or y value that appears next to it.



- By default, when you move a point, its position on the x axis is rounded to the nearest integer. To disable this, you can right-click on the Curve Editor and select **Edit** > **Frame Snap**. You can also momentarily disable the snapping by pressing **Shift** while moving a point.
- To move several points at the same time, select them and drag the selection box to a new location.



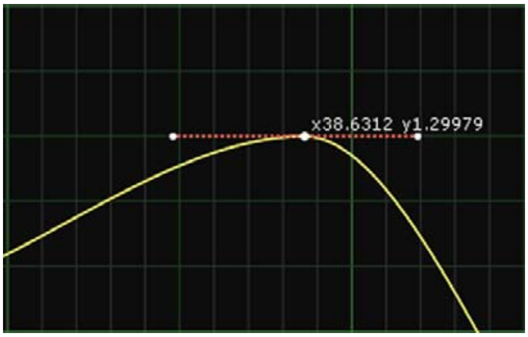
To add or remove points to or from the selection box, **Shift**+click on the points.

To resize and scale the selection box, drag its edges. If the selection box is very narrow, you can press **Ctrl/Cmd** when resizing it. This allows you to resize the box in one dimension only. For example, if you have a box that's wide on the x axis but flat on the y axis, you can resize it in this way along the x axis.

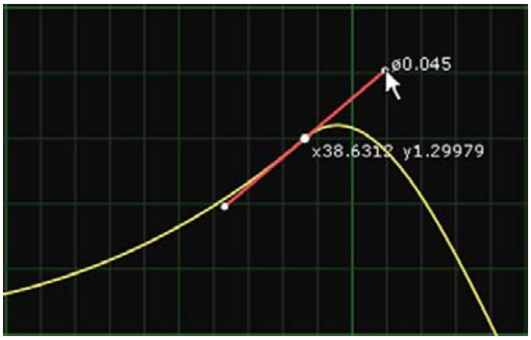
To avoid accidentally moving a point inside the selection box, press **Ctrl/Cmd+Shift** when dragging the box to hide the points inside the box.

To adjust the slope around the points:

1. Select a point on the curve. Red tangent handles appear on both sides of the point.

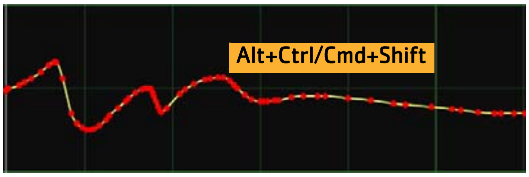


2. Drag the tangent handles to a new location. The curve follows the handles.



To sketch a curve freely:

Press **Alt+Ctrl+Shift** (Mac users press **Alt+Cmd+Shift**) while drawing a curve on the editor. Nuke sketches a curve that follows your mouse movements.



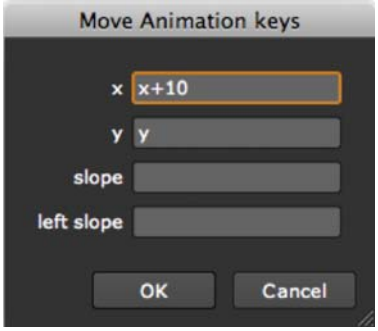
To cut, copy, and paste selected points, expressions, or curves:

1. Right-click on the Curve Editor.
2. From the menu that opens, select **Edit** and the editing function you want to use on the entire curve, for example:
   - **Edit** > **Copy** > **Copy Selected Keys** to only copy the points you have currently selected.
   - **Edit** > **Copy** > **Copy Curves** to copy an entire curve.

- **Edit** > **Copy** > **Copy Expressions** to copy the expression that creates the curve.
- **Edit** > **Copy** > **Copy Links** to copy a curve and keep its values linked to the original curve, so that if you change the original, your changes also affect the copied curve.

To move selected points on the curve by a fixed value:

1. Select all the points you want to move.
2. Right-click on the editor and select **Edit** > **Move**. The *Move Animation Keys* dialog opens.
3. In the **x** and **y** fields, define how you want to move the points along the x and y axes. For example, to shift the selected points to the right by a value of 10, enter **x+10** in the x field.

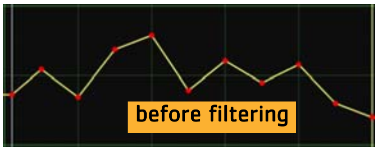


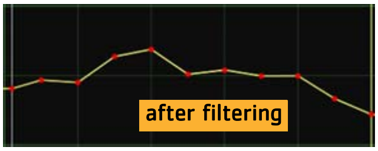
x x+10
y y

4. In the **slope** and **left slope** fields, define how you want to move the points' tangent handles.

To smooth the curve with filtering:

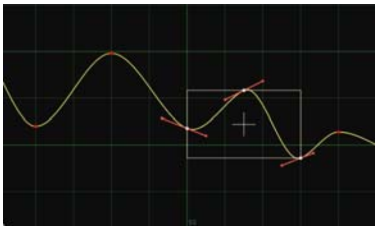
1. Select the portion of the curve that needs smoothing.
2. Right-click on the editor and select **Edit** > **Filter**. The *Filter Multiple* dialog opens.
3. In the **No. of times to filter** field, specify how many times you want to filter the curve. Filtering sets new values on each point based on the average values of their neighbouring points. The more filtering, the smoother the curve.

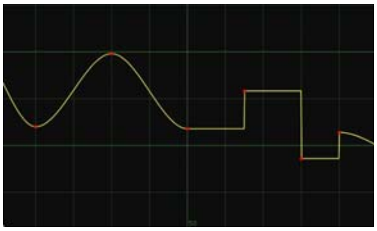




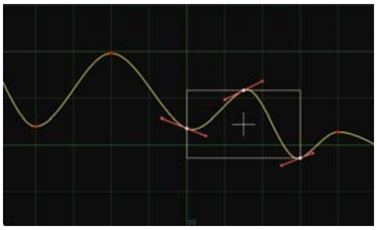
To interpolate parts of a curve:

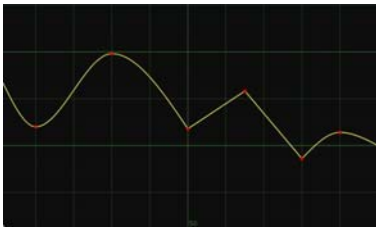
1. Select the point(s) between or around which you want to interpolate the curve.
2. Right-click on the Editor. Select **Interpolation** and the type of interpolation you want to use. Select
   - **Constant** to force a constant value after each selected point.



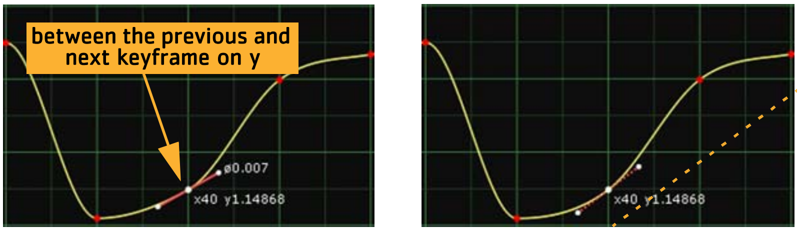


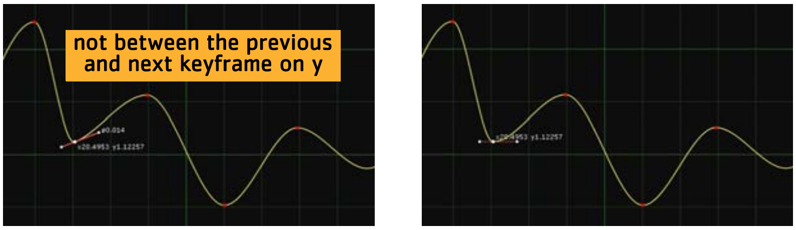
   - **Linear** to use linear interpolation. This produces sharp changes at key frames and straight lines between them.



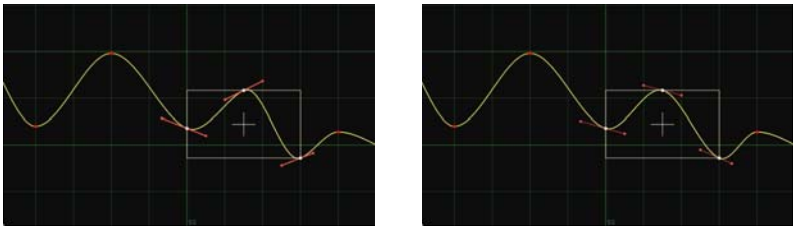


   - **Smooth** to set the tangents' slopes equal to the slope between the keyframe to the left and the keyframe to the right if the selected point is between these two key frames along the y axis. If the selected point is not between these key frames and has a larger or smaller value than both key frames, the tangents' slopes are made horizontal. This ensures the resulting curve never exceeds the keyframe value.

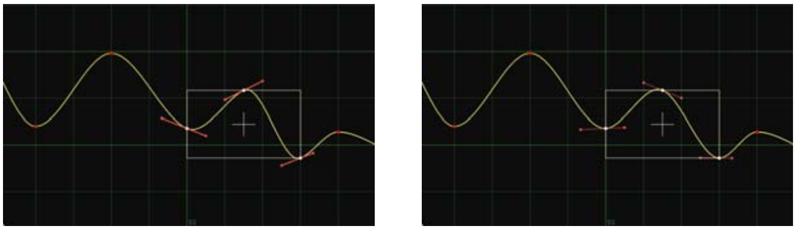

between the previous and next keyframe on y
ø0.007
x40 y1.14868
x40 y1.14868


not between the previous and next keyframe on y
ø0.014
x20.4953 y1.12257
x20.4953 y1.12257

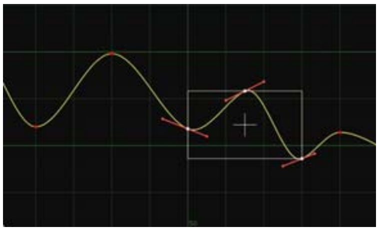
- **Catmull-Rom** to set the tangents’ slope equal to the slope between the keyframe to the left and the keyframe to the right regardless of where the selected point is located. The resulting curve can exceed the keyframe values.

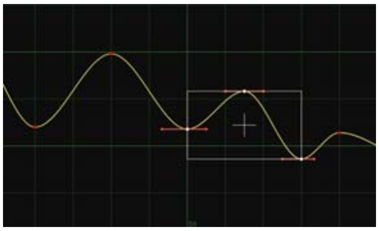


- **Cubic** to set the slope so that the second derivative is continuous. This smoothens the curve.

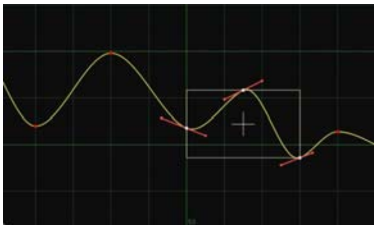


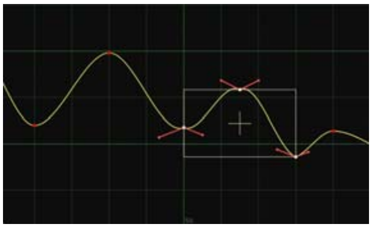
- **Horizontal** to make the tangents horizontal, setting the slope around the selected points to zero.



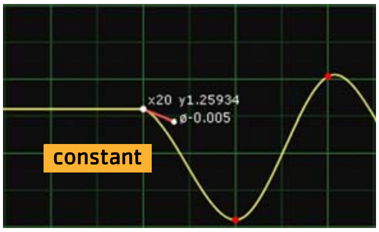


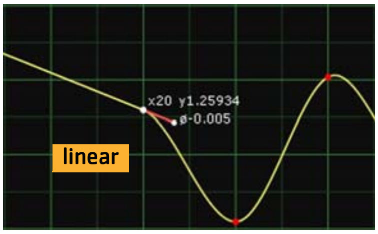
- **Break** to adjust the two tangents of a selected point independent of each other.



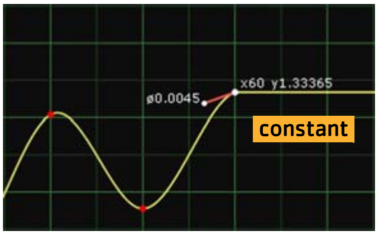


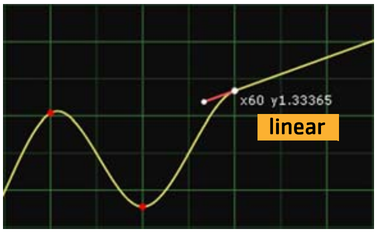
- **Before** > **Constant** or **Linear** to interpolate the parts of the curve that are on the left side of the first point. This option only works if you have selected the first point on the curve.


x20 y1.25934
ø-0.005
constant


x20 y1.25934
ø-0.005
linear

- **After** > **Constant** or **Linear** to only interpolate the parts of the curve that are on the right side of the last point. This option only works if you have selected the last point on the curve.


ø0.0045
x60 y1.33365
constant

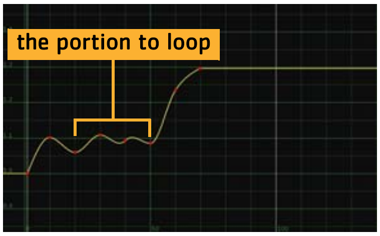

x60 y1.33365
linear

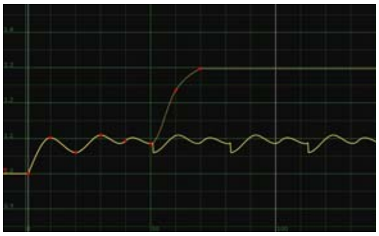
To repeat a portion of the curve throughout the curve:

1. Right-click on the editor and select **Predefined** > **Loop**. The *Loop* dialogue opens.

2. In the **First frame of loop** field, enter first frame of the portion you want to repeat throughout the curve.
3. In the **Last frame of loop** field, enter the last frame of the portion you want to repeat.
4. Click **OK**.

   The shape of the curve between these frames is repeated throughout the rest of the curve. The solid line represents the actual curve, and the dotted line the original curve with the key frames.

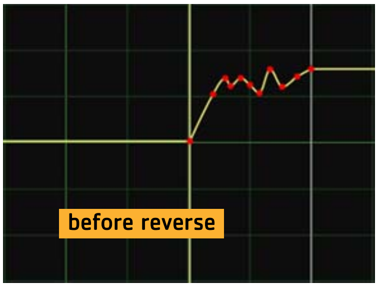


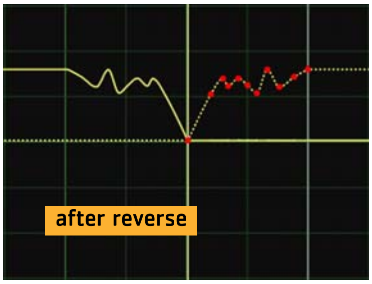


To reverse a curve:

Right-click on the editor and select **Predefined** > **Reverse**.

This makes the curve go backward in time. Both the new curve and the original curve are displayed. The solid line represents the actual curve, and the dotted line contains the key frames that you can modify.

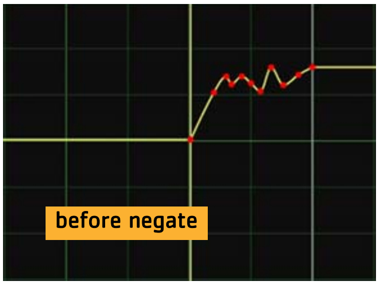


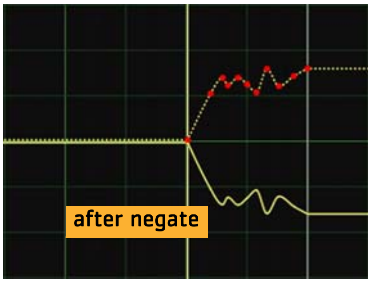


To negate a curve:

Right-click on the editor and select **Predefined** > **Negate**.

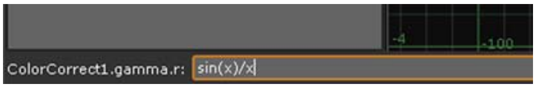
The curve becomes the negative of the key frames. For example, a value of 5 turns into -5. Both the new curve and the original curve are displayed. The solid line represents the actual curve, and the dotted line contains the key frames that you can modify.





To use an expression to modify a curve:

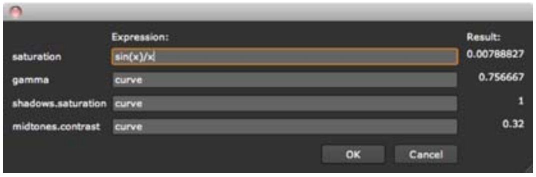
Enter the expression in the expression field at the bottom of the Curve Editor,



ColorCorrect1.gamma.r: sin(x)/x

OR

1. Right-click on the Editor, and select **Edit** > **Edit expression**.
2. In the dialog that opens, type the expression you want to use for the curve, for example, **sin(x)/x**.



	Expression:	Result:
saturation	sin(x)/x	0.00788827
gamma	curve	0.756667
shadows.saturation	curve	1
midtones.contrast	curve	0.32

3. Click **OK**.

Viewers

Viewer nodes, unlike process nodes, don't alter data in any way; rather, they act as windows on it. Each Viewer node displays the render output of any connected process nodes in the Viewer panel. Viewer nodes are essential for quickly assigning the right values to parameters because they allow you to edit in context—that is, edit a given node's parameters upstream in a script while viewing the effect of those changes downstream.

You can place as many Viewer nodes in a script as you wish, which allows you to simultaneously view multiple outputs. You can also pipe the output from up to ten process nodes into single Viewer node, and then cycle through the various displays. This allows you to easily compare an image before and after processing by a given effect.

Note *If you're using the 32-bit version of Nuke, the maximum image size the Nuke Viewer can display is 2^28 = 268,435,456 pixels. This is the same as 16k x 16k, 32k x 8k, 64k x 4k, or 128k x 2k. If your image is larger than this, it will be resized and you will get the following warning:*

Viewer image is clipped to <size> x <size>!

For example, if your image resolution is 60,000 x 4473, Nuke is able to display the image because the number of pixels is less than 2^28. However, if the resolution is 60,000 x 4474 (more than 2^28 pixels), the image will be resized to 59998 x 4474.

In addition to the Viewer, this limit is also applied to the bounding box of the images being passed between each node.

If you're using the 64-bit version of Nuke, however, the maximum image size the Viewer can display is 64k x 64k (or the equivalent number of total pixels at other resolutions). Make sure though, that you have sufficient RAM memory available if you want to use the maximum image size.

Adding Viewer Nodes

Viewers have corresponding nodes that appear in the Node Graph. These nodes do not produce output for rendering; they generate display data only. You can connect Viewer nodes as described in "Working with Nodes" on page 46. In practice, you'll work faster by using the Viewer hotkeys described below.

To add a Viewer node:

1. Select the node whose output you wish to view.
2. Do one of the following:
   - Using the menu bar, choose **Viewer** > **Create New Viewer**.
   - Using the Toolbar, choose **Image** > **Viewer**.
   - Using a keyboard shortcut, press **Ctrl+I** (Mac users press **Cmd+I**).

   Nuke connects a Viewer node to the node you selected in step 1, and displays the output of the node in the Viewer panel. You can also insert a Viewer node and set up its first connection by simply pressing **1** over the Node Graph.

Connecting Viewer Nodes

Once you add a Viewer node to the script, you can quickly pipe any process node's output to it simply by selecting the process node then pressing any number key. Doing so pipes the output to one of the ten input ports available on every Viewer node (the **0** key represents the tenth slot).

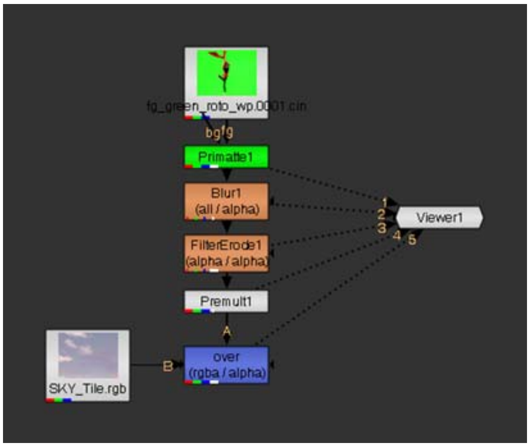

fg_green_roto_wp.0001.cin
bg fg
Primatte1
Blur1
(all / alpha)
FilterErode1
(alpha / alpha)
Premult1
A
SKY_Tile.rgb
B
over
(rgba / alpha)
1
2
3
4
5
Viewer1

Figure 3.21: A Viewer node with multiple inputs.

Toggling Views

If a Viewer node has multiple inputs, like the one depicted above, you can press the up or down arrow keys to quickly cycle through the views (your cursor needs to be in the Viewer window). To view a particular node press the number key (**1, 2, 3... 0**) corresponding to the pipe number whose contents you wish to view.

Panning and Zooming the Viewer Window

To pan the frame:

Hold the middle mouse button and drag on the display (you can also use **Alt**+drag). The frame follows the mouse pointer.

Note *On Linux, **Alt**+drag may not work as expected. This is due to the default window functionality on Gnome. To get around it, you can use the **Windows** key instead of **Alt** when panning.*

Alternatively, you can change your window preferences on Gnome to fix the problem:

1. *Select **Applications** > **Preferences** > **Windows** to open the Window Preferences dialog.*
2. *Under **Movement Key**, select **Super (“or Windows logo”)**.*

   *You should now be able to pan with **Alt**+drag.*

To recenter the frame:

Click the middle mouse button or press **F**.

To zoom in on the frame:

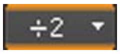
1. Move your pointer over the area of the display on which you want to zoom.
2. Drag the mouse while pressing the middle mouse button and the left mouse button.

   OR

   Press the plus button (+) repeatedly until the frame attains the desired scale.

   OR

   Select **zoom in** from the zoom pulldown menu in the top right corner.



To zoom out from the frame:

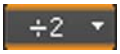
1. Move your pointer over the area of the display from which you want to zoom.
2. Drag the mouse while pressing the middle mouse button and the left mouse button.

   OR

   Press the minus button (-) repeatedly until the frame displays at the desired scale.

   OR

   Select **zoom out** from the zoom pulldown menu in the top right corner.



To restore the zoom to 100%:

Press **Ctrl+1** (Mac users press **Cmd+1**).

Hiding Floating Viewers

To hide a floating Viewer:

Press ` (the accent key).

To show a hidden floating Viewer:

Press ` (the accent key) again.

Using the Viewer Controls

A Viewer's on-screen controls let you navigate the timeline, display channels, zoom, choose cameras (3D mode), and create display wipes and composites.

Timeline controls

Drag the orange marker along the timeline to quickly cue to a specific frame. The number of the current frame appears in the **Current** field above the timeline. You can also cue to a frame by typing its number directly into this field.

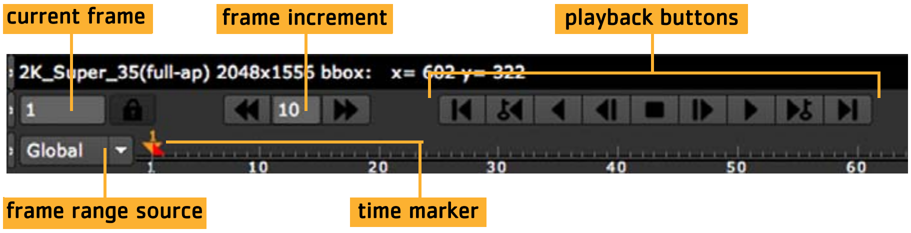


Figure 3.22: Timeline controls.

By default, Nuke automatically adjusts the timeline of every Viewer window to show the frame range defined in your Project Settings. If no frame range is defined, the frame range of the first image you read in is used as the global frame range.

Viewer timeline controls also have a frame range source pulldown menu that you can use to define where the timeline gets its frame range from. You can set this menu to **Global**, **Input**, or **Custom**. **Global** is the default setting described above.

To have the Viewer adjust the timeline to show the "in" and "out" frames of the current input clip, select **Input** from the frame range source menu. The number of the first frame in the clip is shown in the left end of the timeline and the number of the last frame in the right end. If you change the input of the Viewer, the frame range on the timeline is adjusted accordingly.

To manually adjust the frame range for the current Viewer window, pan and zoom on the timeline until you see the desired frame range and **Custom** becomes selected in the frame range source menu. **Alt**+drag to pan, and **MMB**+drag to zoom in. You can also zoom in on or out of the timeline using the mouse wheel. To reset the zoom, press the middle mouse button over the timeline.

To adjust the playback range for the current Viewer window, **Ctrl**+drag (Mac users **Cmd**+drag) the red playback range marker on the timeline to a new "in" and "out" frames as shown in Figure 3.23, or click on the frame range lock button and enter a new playback range in the playback range field.

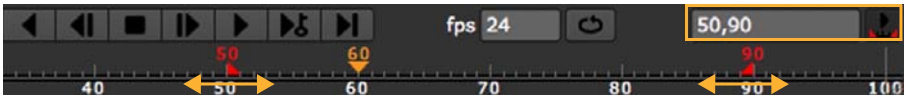


Figure 3.23: Adjusting the frame range for the current Viewer.

To toggle between the new playback range and the visible timeline range, click the button next to the playback range field.

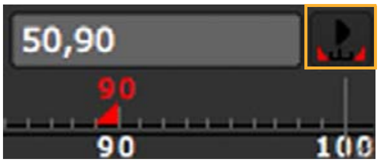


Figure 3.24: Playback is locked to the range defined in the playback range field.

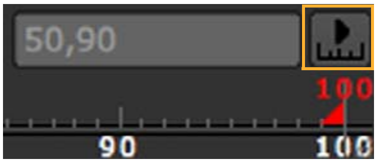


Figure 3.25: The entire visible timeline range is used.

The **fps** field (frames-per-second) initially displays the project's playback speed. Nuke will attempt to maintain this speed throughout playback, although this adjusts depending on the resolution of the imagery and your hardware configuration.

The Frame Increment field lets you specify the number of frames by which the Previous increment/Next increment buttons cue the sequence.

The following table lists the functions of the playback buttons:

Buttons	Functions
⏮ ⏭	The First frame and Last frame buttons cue the sequence to the first and last frame.
◀ ▶	The Previous keyframe and Next keyframe buttons cue the sequence to the script's previous or next keyframe.
◀ ▶	The Play backward and Play forward buttons play the sequence backward or forward at the script's frame rate.
◀| |▶	The Back 1 Frame and Forward 1 Frame buttons cue the sequence to the previous or next frame.

Buttons	Functions
■	The Stop button halts playback.
◀◀ 10 ▶▶	The Previous increment and Next increment buttons cue forward or back by 10 frames by default. These are useful for heavy keyframing tasks. You can adjust the increment value as necessary.

The Playback Mode button lets you control how many times and in what direction the Viewer plays back the sequence. Click the button to toggle between the following modes:

Button	Function
↻	Repeatedly play the sequence (loop).
→|	Play the sequence once from the current frame to the head or tail (stop).
⇄	Repeatedly play the image back and forth from head to tail.

Jumping to a specific frame

You can move quickly to a specific frame on the timeline by choosing **File** > **Go to frame** (or by pressing **Alt**+**G**), entering a frame number in the dialog that appears, and clicking **OK**.

Synchronizing Viewer playback

The Lock/Unlock button lets you toggle synchronized playback of Viewer windows. By default, all Viewers are locked—that is, if you cue to a frame in one Viewer, all other Viewers follow suit.

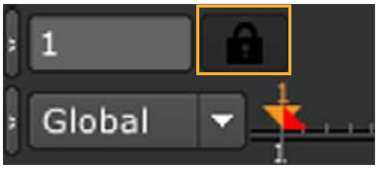
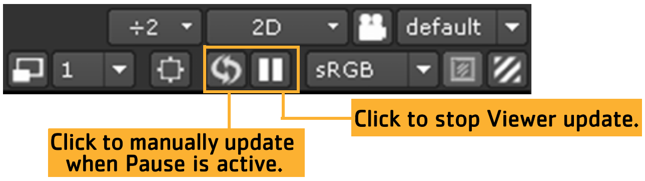


Figure 3.26: Synchronizing Viewers.

When the lock icon changes from a closed lock to an open lock, that Viewer's playback becomes independent of other Viewers, and not cued to the other Viewers.

Pausing the display

The Pause button stops the Viewer from updating and holds the last frame rendered. To reactivate display rendering for all frames, press the button again.



Click to stop Viewer update.

Click to manually update when Pause is active.

Figure 3.27: Pausing the display update of a Viewer.

You can click the render update button next to Pause (or press **U**) to manually update the display while keeping Pause active.

Displaying a single channel

You can press the **R**, **G**, **B**, and **A** keys on your keyboard to display the red, green, blue, and alpha channels respectively. Or, you can also select a channel from the **RGB** pulldown menu in the top left corner.

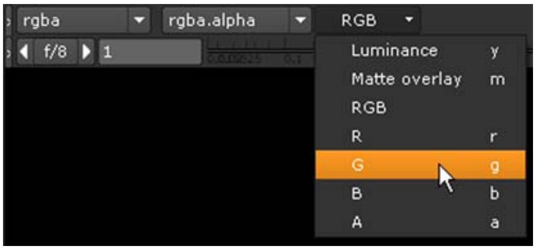


Figure 3.28: Displaying a single channel.

Press one of the channel keys again to toggle back and display all channels.

Tip *If you press **Shift** while selecting the channel, your selection only affects the currently active input of the Viewer node. This way, you can display different channels from the Viewer's different inputs. For example, when keying it can be useful to view the RBG channels from one input and the alpha channel from another, and toggle between the two. To achieve this, do the following:*

1. Create a Viewer with several inputs. See "Connecting Viewer Nodes" on page 88.

2. Activate one of the inputs by pressing its number (for example 1) on the Viewer.

*3. Press **Shift** and select **RGB** from the channel menu.*

4. Activate another input (for example, input 2) by pressing its number on the Viewer.

*5. Press **Shift** and select **A** from the channel menu.*

6. Toggle between the inputs by pressing their numbers or the up and down arrow keys.

Channel Set and Channel Display Lists

The channel set list lets you choose a set of color channels to display in the Viewer. By default, this is set to display the rgba set, but you can choose any channel set in the data stream.

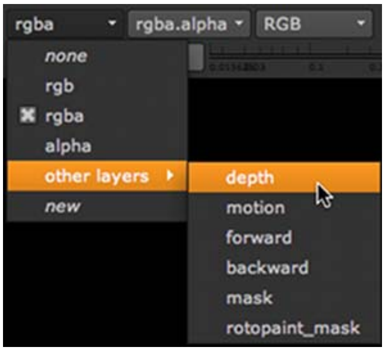

rgba
rgba.alpha
RGB
none
rgb
rgba
alpha
other layers
new
depth
motion
forward
backward
mask
rotopaint_mask

Figure 3.29: The channel set and channel display lists.

The Channel list controls which channel appears when you view the "alpha" channel. The default setting actually does display the alpha channel when you press the **A** key; however, you can change this by selecting any channel in the data stream.

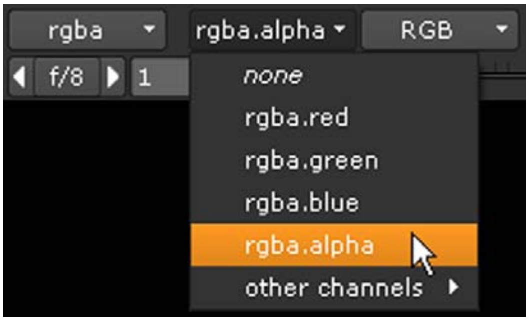

rgba
rgba.alpha
RGB
f/8
1
none
rgba.red
rgba.green
rgba.blue
rgba.alpha
other channels

Figure 3.30: Selecting the channel to display when **A** is pressed.

Superimposing an image's alpha channel over its RGB channels

When you've read in an image that has an alpha channel, you can display the alpha channel as a red overlay on top of the image's red, green, and blue channels.

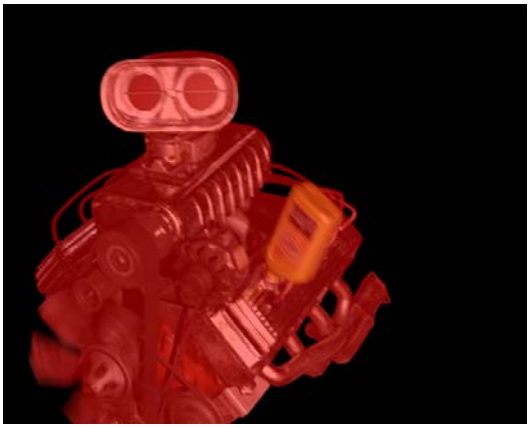
To display an image's alpha channel on its RGB channels:

1. Select **Image** > **Read** to read in an image.
2. Connect a Viewer node to the Read node.

   By default, Nuke displays the red, green, and blue channels in the Viewer.

3. Click on the Viewer to make sure it's the currently active panel.
4. Press M.

   Nuke displays the image's alpha channel as a red overlay on top of the RGB channels.



5. To return to the RGB display, press M again.

Image format labels

The Pixel Value indicator displays information about the pixel underlying the pointer or about a sampled pixel or region of pixels. (You can sample a single pixel from the Viewer by pressing **Ctrl/Cmd** while clicking, a region from the Viewer by pressing **Ctrl/Cmd**+**Shift** while dragging, a single pixel from the node's input by pressing Ctrl/Cmd+Alt while clicking, and a region from the node's input by pressing Ctrl/Cmd+Alt+Shift while dragging.) From left to right, the indicator displays the following about the current pixel or sample: its x and y position; its Red, Green, Blue, and Alpha values; and other values depending on the color type you have selected from the color type menu on the right.

The Format indicator displays the image resolution and the size of the bounding box.

Using the Zoom list

The Zoom list lets you select the magnification factor by which the current image is displayed. This list also shows the hotkeys to press to quickly switch between the different zoom settings.

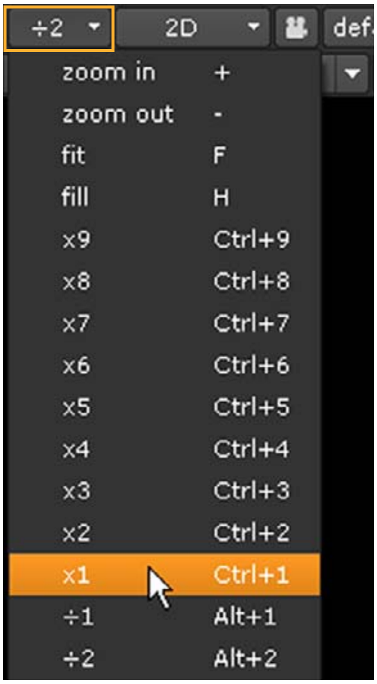


Figure 3.31: Selecting a display zoom.

Proxy mode

Nuke can generate low-res proxies for displayed frames as needed when you press **Ctrl/Cmd+P** or click the proxy mode toggle button on the Viewer to activate the proxy display mode.

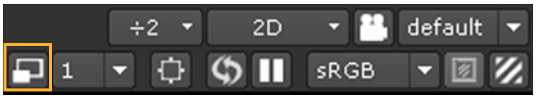
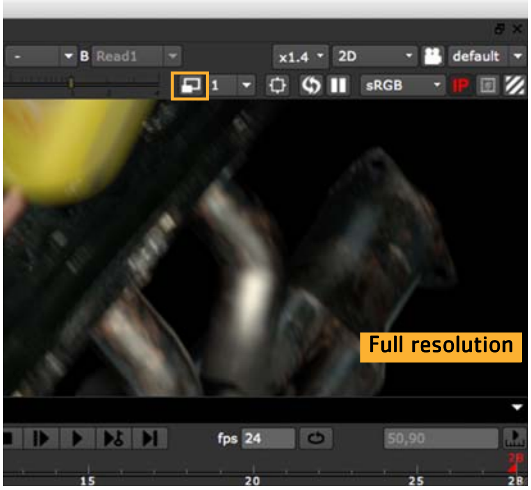

÷2
2D
default
1
sRGB

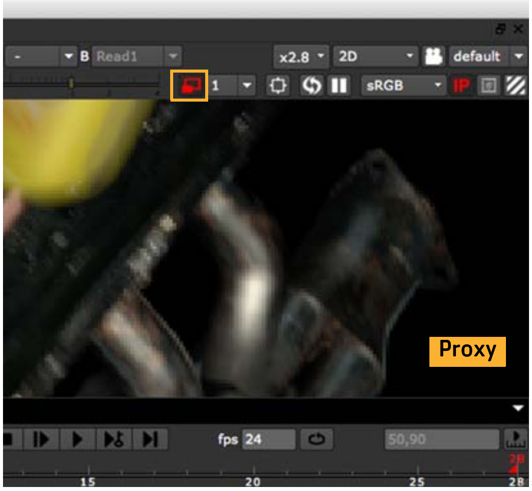
Figure 3.32: Proxy mode toggle.

For more information, "Proxy Mode" on page 124.


B Read1
x1.4
2D
default
1
sRGB
fps 24
15
20
25
28

Full resolution

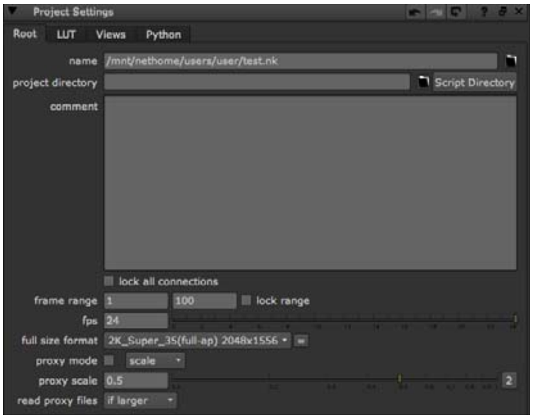
Figure 3.33: High-res display.


B Read1
x2.8
2D
default
1
sRGB
fps 24
15
20
25
28

Proxy

Figure 3.34: Proxy display.

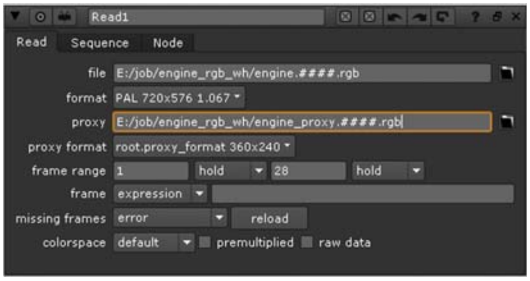
The global proxy resolution and/or scale are determined by the project settings, which you can open by selecting **Edit** > **Project Settings** (or pressing **S**).



name /mnt/nethome/users/user/test.nk
project directory
comment
☐ lock all connections
frame range 1 100 ☐ lock range
fps 24
full size format 2K_Super_35(full-ap) 2048x1556
proxy mode ☐ scale
proxy scale 0.5 2
read proxy files if larger

Figure 3.35: Proxy display resolution defined on the Project Settings properties panel.

You can also read in rendered proxies using the Read nodes' controls. The proxy file does not need to match the proxy resolution in use. Depending on your settings, either the full-res or proxy file will be scaled to the required proxy size. For more information, see "Read nodes and proxy files" on page 126.



file E:/job/engine_rgb_wh/engine.####.rgb
format PAL 720x576 1.067
proxy E:/job/engine_rgb_wh/engine_proxy.####.rgb
proxy format root.proxy_format 360x240
frame range 1 hold 28 hold
frame expression
missing frames error
colorspace default ☐ premultiplied ☐ raw data

Figure 3.36: Reading in proxy versions of images.

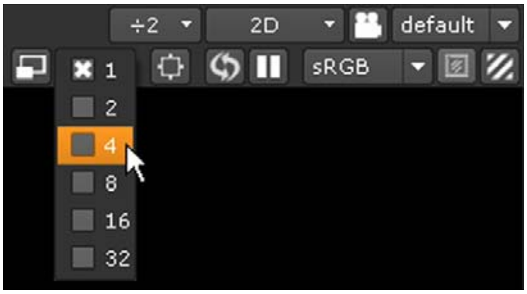
Lowering the display resolution of individual Viewers

Viewers also have a pulldown menu that allows you to easily switch to lower display resolutions, regardless of whether you have activated proxy mode or not. Using this multiplier setting, you can, for example, change the display resolution of an individual Viewer to 50% of the current (be it full-size or proxy) resolution. This is useful if you want to have Nuke display your images more quickly without having to touch the project settings. It also comes in handy if you have just a few very large plates in your script, as you can choose to use lower resolutions when viewing just these plates.

To lower the display resolution of individual Viewers:

From the Viewer's down-rez menu, choose the factor by which you want to lower the display resolution. Select:

- **1** to display 1/1 of the currently active resolution.
- **2** to display 1/2 of the currently active resolution.
- **4** to display 1/4 of the currently active resolution.
- **8** to display 1/8 of the currently active resolution.
- **16** to display 1/16 of the currently active resolution.
- **32** to display 1/32 of the currently active resolution.



For example, if you have a 4K plate and are using a proxy scale of 0.5, your plate will still be 2K even in the proxy mode. Setting the down-rez factor to **2** in the Viewer will scale the plate down further to 50% of the proxy resolution, that is to 1K. This gives you much faster (but less accurate) feedback.

Pixel aspect ratio

The pixel aspect ratio determines whether your images are displayed using square or rectangular pixels. By default, the Viewer uses the pixel aspect ratio defined in your project settings. To see the current setting, select **Edit > Project Settings** (or press **S**).

For example, a pixel aspect ratio of 2 accurately displays anamorphic footage the way it will be projected, as shown in Figure 3.37:

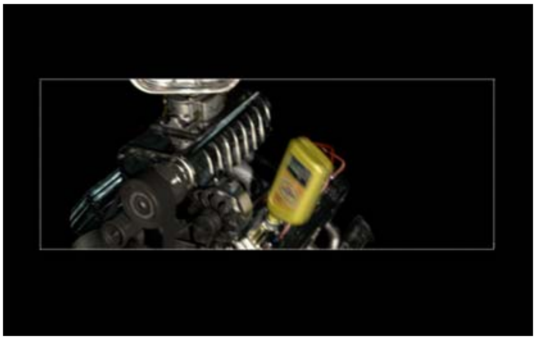


Figure 3.37: The Viewer uses the pixel aspect ratio defined for the script.

If you want to ignore the pixel aspect ratio, you can toggle it by pressing **Ctrl/Cmd+Shift+P** over the Viewer window.

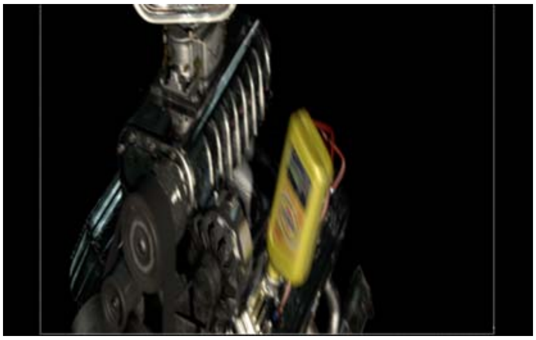


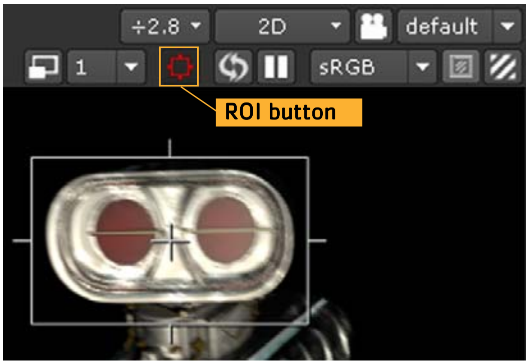
Figure 3.38: Press **Ctrl/Cmd+Shift+P** over the Viewer window to ignore the pixel aspect ratio.

Region of interest (ROI)

The **ROI** button lets you enable rendering only through a region of interest—a portion of the image you explicitly select. This is useful for quickly viewing render results in a process-heavy script.

To define a region of interest:

1. Click on the ROI button in the Viewer controls. The ROI overlay appears.



2. Drag to resize and move the ROI overlay as necessary.

   OR

1. Over the Viewer, press **Alt+W** once (do not hold the keys down). The ROI button turns red, but the ROI overlay does not appear. This allows you to freely draw your own ROI rather than adjust the default overlay.
2. Drag a marquee to draw the region of interest where you need it.

To clear a region of interest:

1. After you've set a region of interest, you can clear it by pressing **Alt+W** over the Viewer. You can then drag a new marquee to define a new region of interest.
2. To turn off the feature and update the whole Viewer with the recent changes, click the **ROI** button again (or press **Shift+W**).

Adjust display gain and gamma

The gain and gamma sliders let you adjust the displayed image, without affecting your final output. These controls are useful for tasks like spotting holes in mattes. You can boost or reduce gain by entering a multiplier (exposure value), dragging on the slider, or using the F-Stop arrows. Boost or reduce gamma by entering a gamma level or dragging the gamma slider.

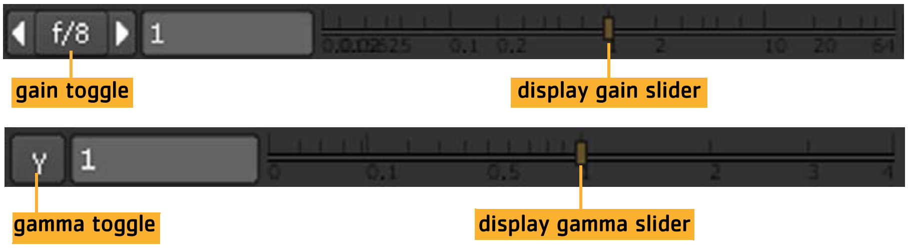


Figure 3.39: Display gain and gamma controls.

The gain and gamma toggle buttons let you switch between the default values of 1 (no change) and the last gain and gamma adjustments you made in the Viewer.

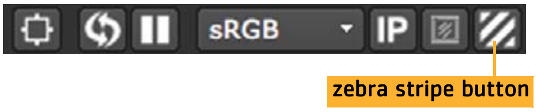


Figure 3.40: The Zebra Stripe button.

Press the Zebra Stripe button to apply stripes to all pixels outside the range 0.0 to 1.0.

Input Process and Viewer Process controls

Input Process and *Viewer Process* operations can be used to modify the image from the viewed node before it is displayed on your monitor. Both only affect the Viewer in which they are activated and do not affect your rendered output. Input Process is a legacy system which uses a node instantiated in the Node Graph to process the image. This is handy for script-specific, temporary, or experimental use, but can be error prone due to the node accidentally being deleted or changed and is limited to a single node. The Viewer Process system was added in Nuke 5.2 and allows a gizmo (or compiled node) to be registered from the Python programming language at start-up. The registered item appears in a menu in the Viewer and the node is instantiated internally within the Viewer when the item is selected so there is no danger of accidental deletion or modification. This also enables multiple Viewer Processes to be registered at different points of start-up (as Nuke works through the NUKE_PATH menu.py files).

The Viewer settings contain an option for the Input Process to be applied before or after the Viewer Process, so the two may be used in conjunction, for instance, with the Input Process applying a projection mask after the Viewer Process applies a film look profile. While you could combine the two into a single Viewer Process node, it can be advantageous to keep operations separated. Having both the Viewer Process and Input Process available provides a great deal of flexibility.

You can create an Input Process by creating a node in the Node Graph and naming it as an Input Process using Nuke's **Edit** menu. Once an Input Process has been named, the IP button appears in the Viewer controls. When the IP button is activated, any image you view is passed through the Input Process.

Unlike Input Processes, Viewer Processes are registered using Python. They can be session independent and always appear in the Viewer's Viewer

Process menu. There are two predefined Viewer Processes, **sRGB** and **rec709**, but you can also build and add your own. When a Viewer Process is selected from the Viewer Process menu, any image you view is passed through that Viewer Process.

Whenever possible, the Input Process and Viewer Process are executed on the GPU. 1D LUT and 3D LUT (Vectorfield) have GPU implementations, so the built-in Viewer Processes will run on the GPU (unless **gl buffer depth** has been set to **byte** in the Viewer settings, in which case all processing is done on the CPU). To get the GPU'd versions of the nodes for use in a custom Viewer Process gizmo, press **x** over the Node Graph, enter **ViewerGain**, **ViewerGamma**, or **ViewerClipTest** in the command entry window, and press **Return**.

The following table lists the differences between an Input Process and a Viewer Process.

Input Process	Viewer Process
Set by selecting the node in the Node Graph and choosing **Edit** > **Node** > **Use as Input Process**.	Registered using Python.
Activated using the IP button in the Viewer controls.	Activated using the Viewer Process menu in the Viewer controls.
Requires that the node exists in the Node Graph. Can quickly and easily be modified by artists. Can also be accidentally deleted, disabling the effect.	Is defined in a text file called menu.py that is run at start-up. Accessible for artists, but not likely to be accidentally modified or deleted.
Script dependent. Unless your Input Process node is saved in the template.nk file that is loaded at start-up, the Input Process is lost when you restart Nuke.	Session independent. The Viewer Processes registered in menu.py will always be available in each new session of Nuke.
There can only be one Input Process at a time. Setting a new Input Process overrides any previously used Input Process.	There can be an unlimited number of Viewer Processes available in the Viewer Process menu. For example, it is possible to register Viewer Processes in any menu.py file at start-up, so Viewer Processes can be added at any directory in your NUKE_PATH.
Useful for temporary or non-critical viewing options that you want in the current shot for convenience, or for testing Viewer Processes before registering them. Can also be used for other things, such as field charts or masks that may be switched on or off and changed around in the shot.	Useful for viewing options that you often need or that should not be modified by artists on a shot-by-shot basis.

Note *Note that Input Processes and Viewer Processes are part of a built-in, fixed pipeline of nodes that are applied to images before they are displayed in the Viewer. This pipeline is either:*

gain > Input Process > Viewer Process > gamma > dither > channels > cliptest (if ***viewer input order*** *has been set to* ***before viewer process*** *in the Viewer settings)*

OR

gain > Viewer Process > Input Process > gamma > dither > channels > cliptest (if ***viewer input order*** *has been set to* ***after viewer process*** *in the Viewer settings).*

However, depending on what the Input Process and Viewer Process are doing, the order in the built-in pipeline may not be the correct order. Therefore, if your Input Process or Viewer Process have controls that also exist for the Viewer, such as float controls named ***gain***, ***gamma***, *or* ***cliptest***, *then the Viewer will drive them from the corresponding Viewer controls and not do that image processing itself. This allows you to implement these controls in your Input Process or Viewer Process node/gizmo using whatever nodes and order you want. If your Input Process and Viewer Process do not have these controls, then the Viewer will apply the effects in its normal way according to the built-in pipeline.*

In the built-in pipeline, dither is applied to diffuse round-off errors in conversion of floating point data to the actual display bit depth. Although the cliptest is drawn at the end, it is computed on the image as input to the Viewer.

Note *By default, the predefined Viewer Processes,* ***sRGB*** *and* ***rec709***, *affect all channels. However, if you want them to only affect the red, green, and blue channels, you can activate* ***apply LUT to color channels only*** *in the individual Viewer Settings or on the* ***Viewers*** *tab of the preferences.*

Input Process controls

To activate or deactivate the effect of an Input Process, click the IP button in the Viewer controls. Note that the IP button only appears if the **input process** field in the Viewer settings is not empty. The button is also only enabled when a node in the Node Graph is set as an Input Process.

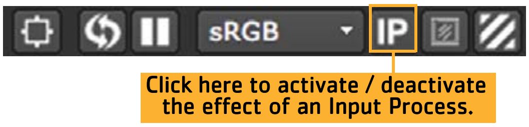

sRGB
IP
Click here to activate / deactivate the effect of an Input Process.

Figure 3.41: Toggling the use of the Input Process.

To open the Viewer settings, press **S** on the Viewer, or select **Viewer Settings** from the Viewer's right-click menu. By default, **input process** is set to **VIEWER_INPUT**. If a node called VIEWER_INPUT exists in the Node Graph, it will automatically be used as the input process for the Viewer. This ensures backwards compatibility with pre-5.2 scripts.

However, the Input Process node does not have to be named VIEWER_INPUT. You can use any node as an Input Process. Do the following:

1. Select the node in the Node Graph and choose **Edit** > **Node** > **Use as Input Process**.

   Alternatively, you can press **S** on the Viewer to open the Viewer settings and enter the name of the node in the **input process** field.
2. In the Viewer settings, you can also define whether the Input Process will be applied before or after the Viewer Process currently in use. To do so, set **viewer input order** either to **before viewer process** or **after viewer process**.

The Input Process node should not be connected to other nodes in the Node Graph. If you attempt to connect it, an error will be displayed on the Viewer. If you delete the Input Process node from the Node Graph, the effect of the Input Process will be disabled.

Viewer Process controls

To activate a Viewer Process, select it from the Viewer Process menu in the top right corner of the Viewer. Any images you now view using this Viewer are passed through the selected Viewer Process.

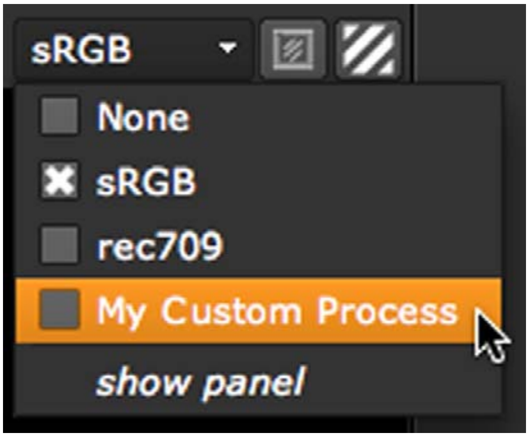


Figure 3.42: Selecting the Viewer Process to apply to images before they are displayed in the Viewer.

Nuke includes the following predefined Viewer Process gizmos: **sRGB** and **rec709**. By default, sRGB is used because it is a best-guess default for a typical computer monitor.

In addition to using the predefined Viewer Processes, you can also add your own by registering a node or gizmo as a Viewer Process. You can register as many Viewer Processes with custom Viewer LUTs as you like. For more information on creating and registering custom Viewer Processes, see "Creating Custom Viewer Processes" on page 651.

All available Viewer Processes (both predefined and custom ones) appear in the Viewer Process menu in the Viewer controls. To disable the use of a Viewer Process, select **None** from the Viewer Process menu.

To open the properties panel of the currently active Viewer Process, select ***show panel*** from the Viewer Process menu.

Note that if the control you want to adjust has the same name as any of the Viewer controls (for example, **gain** or **gamma**), you should adjust the control on the Viewer. This drives the control in the following:

- the Input Process' controls if an Input Process is in use
- in the Viewer Process' controls if no Input Process is in use.

Tip *If you want to render out a file with the Viewer Process effect baked in, you can select **Edit > Node > Copy Viewer Process to Node Graph** to create an instance of the Viewer Process node in the Node Graph.*

Monitor output toggle

The monitor output button lets you preview the current Viewer image on an external broadcast video monitor. You can select the external device and the display mode to use in the Viewer settings. To open the Viewer settings, press **S** on the Viewer.

Note that this option is only available in 32-bit Windows and Mac OS X versions of Nuke, and requires additional hardware, such as a monitor output card or a FireWire port.

For more information, see "Previewing on an External Broadcast Video Monitor" on page 521.

2D / 3D Toggle and Camera Controls

The 2D / 3D list lets you toggle between 2D and 3D display modes in the current Viewer. This list also lets you choose between different orthographic (non-perspective) views when working in the 3D mode.

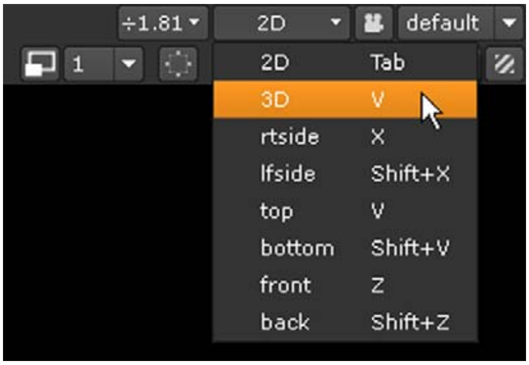


Figure 3.43: The 2D / 3D views and camera controls.

The camera list on the right lets you choose which camera to look through when multiple cameras exist in your 3D scene. For more information on these controls, see Chapter 15, "3D Compositing", on page 426.

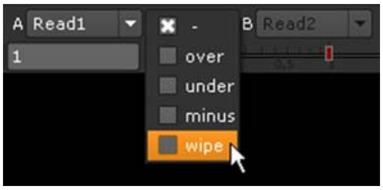
Using the Viewer Composite Display Modes

The wipe control provides an option for displaying a split-screen of two images, which can help you compare before and after versions for color correction, filtering, and other image manipulation. This control also includes display compositing options to overlay different images.

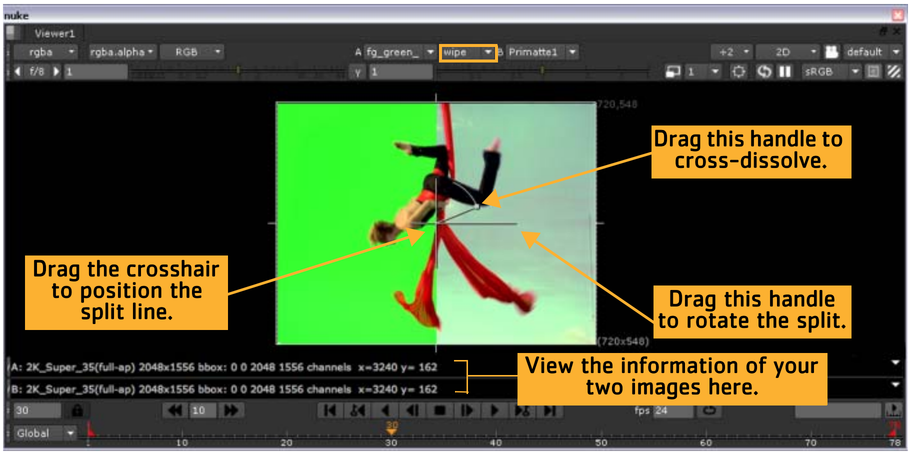
To display a comparison wipe:

1. Select a node in your script and press **1** to display its output in the Viewer.
2. Select the node you want to compare and press **2**.

   The **2** keystroke connects the image to the Viewer (assigning the next available connection, number 2).
3. From the **A** and **B** lists on top of the Viewer, select the images you want to compare. The lists include the last four nodes connected to the Viewer.
4. From the Viewer composite list in the middle, select **wipe**.



The two images are displayed split-screen in the Viewer. You can view their details in the A and B information bars at the bottom of the Viewer.



Drag this handle to cross-dissolve.

Drag the crosshair to position the split line.

Drag this handle to rotate the split.

```
A: 2K_Super_35(full-ap) 2048x1556 bbox: 0 0 2048 1556 channels x=3240 y= 162
B: 2K_Super_35(full-ap) 2048x1556 bbox: 0 0 2048 1556 channels x=3240 y= 162
```

View the information of your two images here.

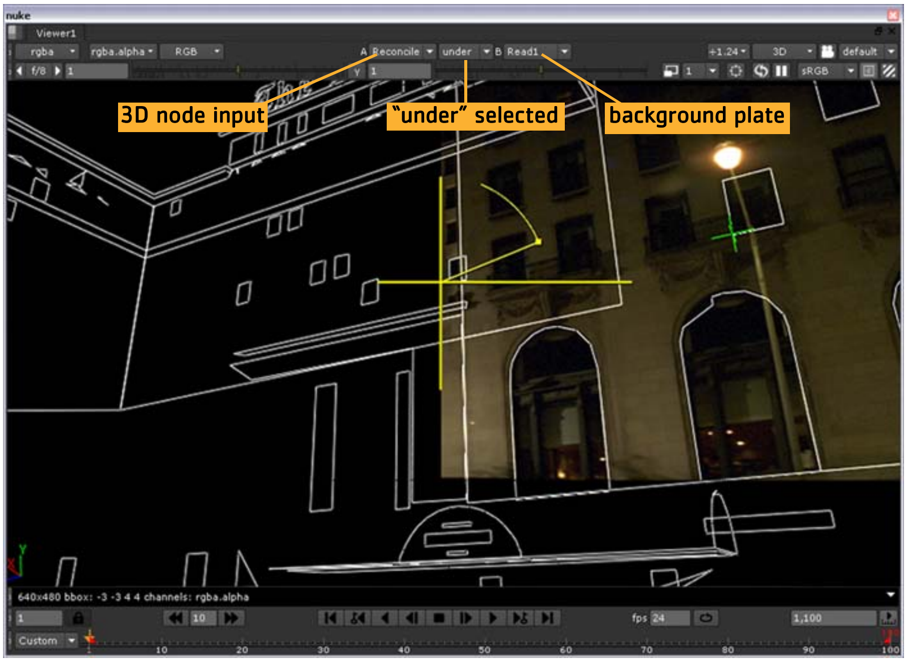
5. Drag the handles of the crosshair to adjust the wipe:
   - Drag the crosshair center to change its position.
   - Drag the long handle (on the right) to rotate the wipe.
   - Drag the "arc" handle to cross-dissolve the second image.
6. When finished with the split-screen, select none (-) from the Viewer composite list.

Tip *If you press **Shift** while selecting a channel, your selection only affects the currently active input of the Viewer node. This way, you can display different channels from the Viewer's different inputs. For example, when keying it can be useful to view the RBG channels from one input and the alpha channel from another, and toggle between the two.*

The display composite options—**over**, **under**, and **minus**—can also be selected to overlay two images. When the two images are 2D, this allows you to create a quick comp.

When one image is 2D and the other is a 3D node, you can use **under** to line up the wireframe preview with the 2D reference, and see how the 3D matches prior to a full render.

One example of this is when you want to preview a wireframe 3D scene with a background plate that you are trying to match, as shown below.


3D node input
“under” selected
background plate
Viewer1
rgba
rgba.alpha
RGB
A Reconcile
under
B Read1
3D
default
sRGB
fps 24
Custom

640x480 bbox: -3 -3 4 4 channels: rgba.alpha

Figure 3.44: Comparing a 3D scene over a 2D image.

For more information, see Chapter 15, "3D Compositing", on page 426.

Hiding and Showing Viewer Toolbars

To hide or show the top toolbar, press {.

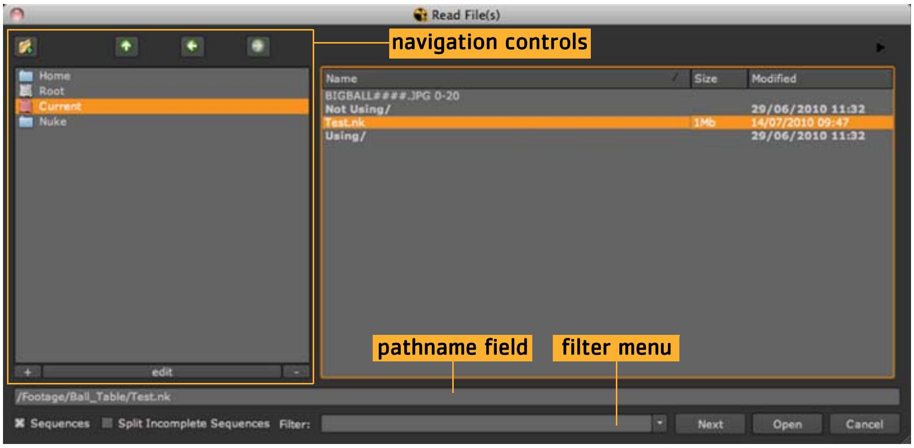
To hide or show the bottom toolbar, press }.

Locking the Viewer Zoom Level

You can choose to lock the Viewer zoom level, so that it doesn’t change when you switch between inputs of different sizes. Right-click on the Viewer, and select **Prevent Auto Zoom** to toggle between maintaining the same zoom level for all inputs (on) and changing it according to on-screen image dimensions (off). Alternatively, you can also press **Alt+Z** on Viewer, or toggle **prevent auto zoom** in the Viewer settings.

Using the File Browser

Whenever you load or save files in Nuke, you'll see a browser similar to the one shown in Figure 3.45. The directory navigation buttons let you create or access the directory from which you wish to read or write data.


Read File(s)
navigation controls
Home
Root
Current
Nuke
edit
pathname field
filter menu
Sequences
Split Incomplete Sequences
Filter:
Next
Open
Cancel

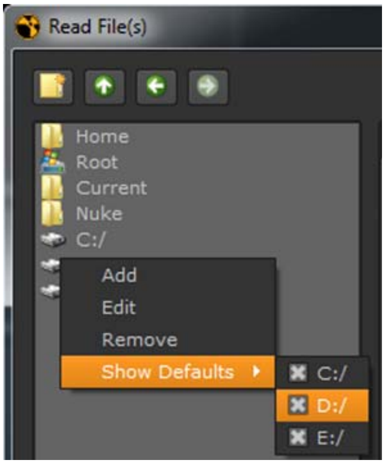
Name	Size	Modified
BIGBALL####.JPG 0-20		
Not Using/		29/06/2010 11:32
Test.nk	1Mb	14/07/2010 09:47
Using/		29/06/2010 11:32

/Footage/Ball_Table/Test.nk

Figure 3.45: Nuke's file browser.

The navigation controls let you move through the directory structure, bookmark favorite directories, and create new directory folders.

Windows only: You can show/hide the drives that Windows auto creates by right-clicking the required drive, selecting **Show Defaults**, and checking or unchecking the drive.


Read File(s)
Home
Root
Current
Nuke
C:/
Add
Edit
Remove
Show Defaults
C:/
D:/
E:/

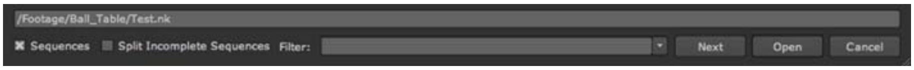
To use the navigation controls:

- Click the **Create New Directory** button to create a new directory at your current position in the file hierarchy.

- Click **Up one directory** to go up one directory closer to the root.
- Click **Previous directory** to go back one directory.
- Click **Next directory** to go forward one directory.
- Click **Home** to access the directory defined as your local working directory.
- Click **Root** to ascend to the very top of your local drive or server's file hierarchy.
- Click **Nuke** to access the directory you (or your system administrator) defined as your network working directory.
- Click the **+** button to add a directory bookmark.
- Click the **edit** button to edit the name or pathname to a bookmark.
- Click the **–** button to remove a directory bookmark.

Pathname field

The pathname field displays the current directory path, lets you navigate to a new path, and also enter a filename for scripts and rendered images.



/Footage/Ball_Table/Test.nk

☒ Sequences ☐ Split Incomplete Sequences Filter:

Figure 3.46: Pathname field.

To use the pathname field:

1. To navigate to a directory, type the pathname in the field.
2. To enter a script name, browse to a directory path and enter the file name after the displayed path.

To limit the file list to specific file types, use the **filter** pulldown menu and **Sequences** checkbox.

To use the filter pulldown menu and Sequences checkbox:

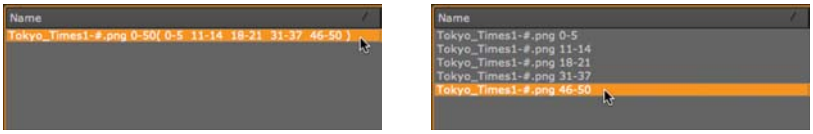
- Select ***.nk** to display only Nuke script files.
- Select ***** to display all files (except hidden files), regardless of whether they're associated with Nuke.
- Select .*** *** to display all files, including hidden files.
- Select ***/** to display directory names, but not their contents.
- Check **Sequences** to display image sequences as single titles, as in fgelement.####.cin 1-50 rather than fgelement.0001.cin, fgelement.0002.cin, fgelement. 0003.cin, and so on.

You can also split incomplete sequences in to separate Read nodes using the **Split Incomplete Sequence** checkbox.

To use the Split Incomplete Sequence checkbox:

- Locate an incomplete sequence.
- Check **Split Incomplete Sequence** and select all the resultant files.
- Click **Open** to open the sequence in separate Read nodes.

  Figure 3.47 shows the same file sequence with **Split Incomplete Sequence** enabled and disabled.



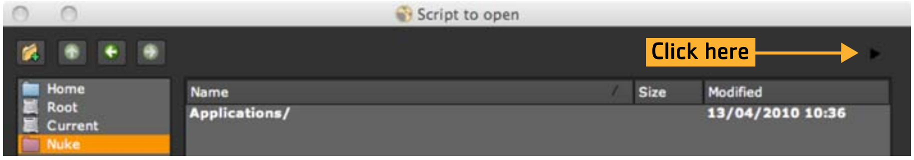
Tokyo_Times1-#.png 0-50(0-5 11-14 18-21 31-37 46-50)

Tokyo_Times1-#.png 0-5
Tokyo_Times1-#.png 11-14
Tokyo_Times1-#.png 18-21
Tokyo_Times1-#.png 31-37
Tokyo_Times1-#.png 46-50

Figure 3.47: Using **Split Incomplete Sequence**.

To preview files in the file browser:

1. Click the black arrow in the top right corner of the file browser.


Script to open
Click here
Home
Root
Current
Nuke
Name
Size
Modified

Applications/ 13/04/2010 10:36

   The file browser expands to include a small viewer.

2. Select the file you want to preview. Nuke displays the file in the file browser.

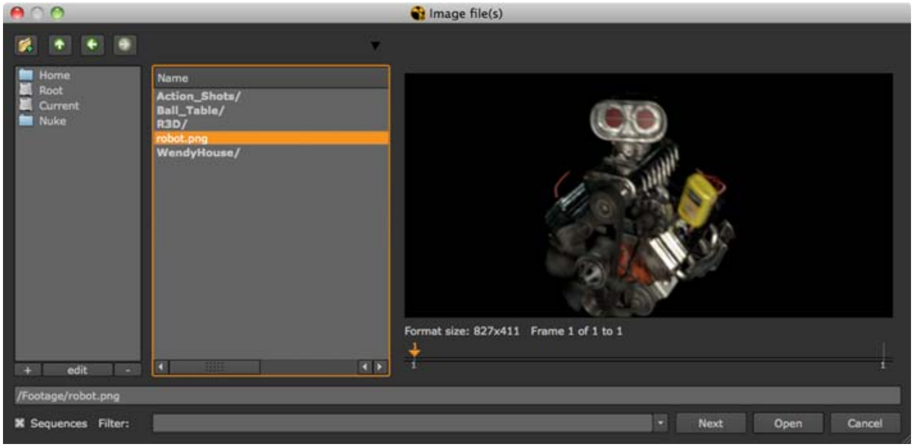


Image file(s)
Home
Root
Current
Nuke
Name
+
edit
-
Sequences
Filter:
Next
Open
Cancel

Action_Shots/
Ball_Table/
R3D/
robot.png
WendyHouse/

Format size: 827x411 Frame 1 of 1 to 1

/Footage/robot.png

To select multiple files with the file browser:

1. Browse to the folder where the files are located.
2. **Ctrl**+click on all the files you want to open to select them (Mac users **Cmd**+click).
3. You can open files from multiple directories by clicking **Next** and browsing to the next file location.

Tip *As you browse around, files that you have previously selected will appear highlighted in the color selected in the* ***Preferences > Appearance*** *tab.*

4. Click **Open**.

   Nuke opens all the files you selected.

To customize file browser colors:

1. In the **Preferences** > **Appearance** tab, select the required **File Browser Colors** for the **Queued Item Color** and **Focus Color** entries.
2. Either:
   - Click **Save Prefs** to retain your settings when you relaunch Nuke, or
   - Click **Close** to use the new settings for this script only.

Undoing and Redoing

Nuke generally gives you an undo history that extends back to the first action of the application's current session.

To undo an action in the workspace:

Select **Edit** > **Undo** (or press **Ctrl/Cmd+Z**). Repeat as necessary.

To redo an action in the workspace:

Select **Edit** > **Redo** (or press **Ctrl/Cmd+Y**). Repeat as necessary.

To undo an change in a properties panel:

Click the **Undo** arrow button in the properties panel.

To redo an change in a properties panel:

Click the **Redo** arrow button in the properties panel.

To undo all changes made after the properties panel was opened:

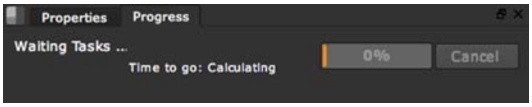
Click the **Revert** button.
OR
Right-click on the properties panel and select **Revert knobs** from the menu that opens.

To set all controls back to their default values:

Right-click on the properties panel and select **Set knobs to default** from the menu that opens.

Progress Bars

Nuke displays a progress bar for each active task it performs. By default, progress bars appear in a pop-up dialog, but you can also display them in the Progress panel. To do so, click on a content menu button and select **Progress Bars**. This opens a Progress panel. The next time you get a progress bar, it will appear in the panel. If you delete the panel, progress bars will appear in a pop-up dialog again.



Waiting Tasks ...
Time to go: Calculating

Figure 3.48: Progress panel.

If you want to have the Progress panel appear in the same position in the future, you can save it as part of a layout. For more information, see “Interface Layouts” on page 116.

Handling Errors

Sometimes things may not go as you planned and you might face an error message in Nuke. When this happens, an error alert displays in the Viewer and on the node that has a problem in the Node Graph. The error message itself prints in the **Error Console** tab next to the Properties Bin. If you have an error in a Read or a Write node, or you are missing a plug-in, the error message also displays in a pop-up window.

If you see an error alert on your node or in the Viewer, you can click the **Error Console** tab to open it and view the error message. If you can’t see the Error Console, click the content menu button and select **Error Console** to display it.

In the Error Console error list, you can double-click on a message and, if possible, Nuke will take you to the control panel of the node that's in error. This isn't always possible because of the nature of the error. You can also click the **clear output** button on the Error Console to clear all error messages on the tab.

Customizing the Interface

You may be used to a certain way of working, or simply disagree with some of Nuke's default settings. If this is the case, you'll be happy to know that you can customize Nuke's interface to find just the layouts and settings that work for you. You can then save your preferred settings, layouts, and template scripts for future use.

Interface Layouts

When your scripts grow, you may not be able to fit all the different elements on your display at the same time. Luckily, you can customize the windows and panes, so that accessing the elements you often need becomes as quick and easy as possible.

Customizing panes

You can resize and split panes to make more room for different elements on the screen.

To resize a pane:

Drag the divider line of the pane into a new location.

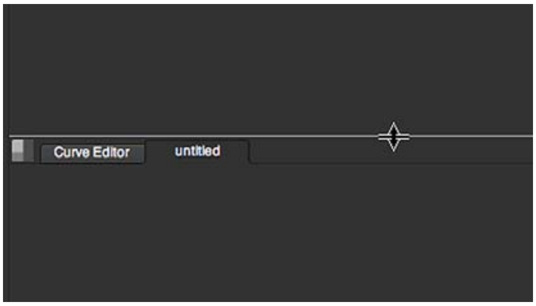


Figure 3.49: Resizing a pane.

To split a pane:

1. Click on the content menu button in the top left corner of the pane.

2. Select **Split Vertical** or **Split Horizontal** from the menu that opens.

Moving the Toolbar

You can move the Toolbar into a new position by adding a new panel for it, hiding the panel name and controls, and resizing the panel. For more information on how to do this, see "Adding tabbed panels" and "Hiding tab names and controls" below.

Adding tabbed panels

When you can't fit more elements to your display, you can use tabs to save space. You can also use tabs to move the Toolbar into a new location.

To add tabs:

1. Click on the content menu button in the top left corner of the pane.
2. Select the type of tab you want to add, for example, **Node Toolbar**, **Node Graph**, **New Viewer**, or **Script Editor**.

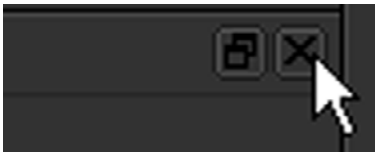
   The new tab is added on top of the existing tabs.

To move tabs:

Click on the name of the tab and drag the tab to a new position inside the same pane or in another pane.

To close tabs:

1. Make sure you are viewing the tab you want to close.
2. Click the "**x**" button in the top right corner of the current tab.



Tabs and floating windows

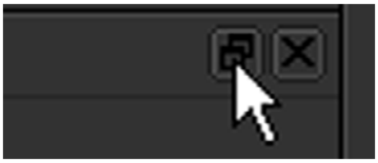
You can turn tabs and panes into floating windows and vice versa.

To turn a tab or pane into a floating window:

1. Make sure you are viewing the tab or pane you want to float.
2. From the content menu, select **Float Tab** or **Float Pane**.

   Alternatively, in the case of tabs, you can also do one of the following:

   • Click the float button in the top right corner of the current tab,



   OR

   • Ctrl+click on the tab name (Mac users Cmd+click),

   OR

   • Right-click on the tab name and select **Float Tab**.

To turn a floating window into a tab or pane:

Click on the tab name or pane in the floating window and drag it to where you want it to dock.

To close floating windows:

Click the "x" button in the top right corner of the tab or pane.

Customizing windows

To make a pane expand to the size of the window:

1. With your cursor in the pane.
2. Press spacebar quickly. (Pressing and holding the spacebar brings up a context sensitive menu for that pane.)

To make a window fullscreen:

1. Make sure the window you want to make fullscreen is active. This could be the main application window or a floating Viewer. Making it fullscreen removes the window borders.
2. Press Alt+S.

Hiding tab names and controls

You can hide the names and control buttons of tabs, as you may not need them with all panels, such as the Toolbar panel.

To hide the names and controls on tabs:

From the content menu, disable **Show Tabs**.

To show the names and controls on tabs:

1. Move your mouse pointer over the very top of the pane area until the top edge of the pane brightens up.
2. Right-click to open the content menu.
3. From the content menu, enable **Show Tabs**.

Saving layouts

You can save up to six favorite layouts and retrieve them as necessary. You can also use these features to setup the Nuke interface for dual monitors.

To save a layout:

1. Open and arrange the Nuke panes, panels, and tabs as desired.
2. Select **Layout** > **Save layout 1 (startup default)** (or press Ctrl/Cmd+F1) to save the default layout. Select **Yes** in the confirmation dialog that appears.

   This step shows F1 as the keystroke, but it can be any key between F1 and F6.

To retrieve a layout:

Select **Layout** > **Restore layout 1(startup default)** (or press Shift+F1) to retrieve the default layout.

F1 is used here, but if you saved a layout to a different function key—any key between F1 and F6—then you can press Shift followed by the key to retrieve the layout.

To use window layout 1 as the default when loading scripts:

1. Select **Edit** > **Preferences** to open the preferences dialog.
2. Go to the **Windows** tab.

3. Check **use window layout 1 when loading scripts**. This is usually checked by default.

4 MANAGING SCRIPTS

In this chapter, you learn about Nuke's project files called *scripts*. The topics covered include setting up, saving, and loading scripts. You'll also learn about managing your node tree in the Node Graph, using Precomp nodes, and working with file metadata.

Working with Multiple Image Formats

Nuke supports multiple file formats, such as Cineon, TIFF, OpenEXR, HDRI, and RAW camera data (via the dcraw command-line program), and allows you to mix them all within the same composite. By default, Nuke converts all imported sequences to its native 32-bit linear RGB colorspace. You can, however, use the Colorspace node to force one of several color models, including sRGB, Cineon, rec709, gamma 1.80/2.20, HSV, or HSL. The Log2Lin node lets you convert between logarithmic and linear colorspace (and vice-versa).

There are no restrictions on image resolution—you can freely mix elements of any resolution within the same script. You can, for example, use a 2k film plate as the background for video shot in PAL format, and then output the result in HD1080i. Nuke automatically adjusts its Viewer to accommodate the image you're viewing.

8-, 16-, and 32-Bit Image Processing

Some digital compositing systems, especially those geared for video work, are optimized for processing exclusively 8-bit elements (that is, images with 256 intensity values per channel). Other systems allow for the mixing of 8, 16, and 32-bit elements.

For Nuke, which began as a film effects tool, image quality is paramount. Thus, it supports the processing of exclusively 32-bit-per channel elements (Elements with lower bit depths are upconverted to 32 bits per channel upon import.) Thirty-two bit support allows for a much richer palette of colors and floating point precision in all script calculations. In practice, this means that Nuke carries out every operation—from an increase in gamma to a transform—with much greater accuracy than a lower-bit-depth system.

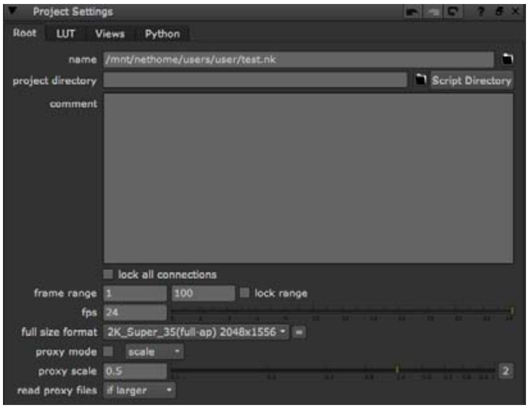
You might expect that this boost in image quality would result in big sacrifice in speed. This is not the case for a couple of reasons. First, Nuke can process scripts with great efficiency because they are uniformly in a 32-bit colorspace. Moreover, hardware manufacturers now routinely design their CPUs for 32-bit processing, making them a efficient match for 32-bit-per-channel images.

Setting Up Your Script

When you start working on a script, you should first define the settings for it. This involves assigning the script a name, frame range, frame rate, and default full and proxy resolution format.

Name, Timespan, and Frame Rate

1. Select **Edit** > **Project Settings**, or simply press **S** over a blank portion of the workspace. The Project settings properties panel appears.



name /mnt/nethome/users/user/test.nk
project directory
comment
☐ lock all connections
frame range 1 100 ☐ lock range
fps 24
full size format 2K_Super_35(full-ap) 2048x1556
proxy mode ☐ scale
proxy scale 0.5
read proxy files if larger

2. On the **Root** tab, type a name for the script (say, **firstcomp.nk**) in the **name** field. Nuke's scripts always have the extension .nk.
3. Type the numbers of the first and last frames in the **frame range** fields to define length of time for your "shot."
4. In the **fps** field, enter the rate in frames per second (fps) at which you want your script's Viewers to play back footage. For film-based elements, 24 fps is appropriate.

Setting the Default Project Directory

In the Project Settings (**Edit** > **Project Settings**) you can define your default project directory. You can then refer to it with './' in your file paths, for example **./test.jpg** for an image called "test" in your default project directory.

You can also click the **Script Directory** button to create an expression that will set your **project directory** to be the directory where your current script is saved.

Full-size Formats

When you start a new project in Nuke, you need to set up a full-size format for the project. The full-size format determines the size of the "black" image that you get from any disconnected node inputs. It also sets the default size of any script-generated elements, such as Constants, Colorbars, Bezier and B-spline shapes.

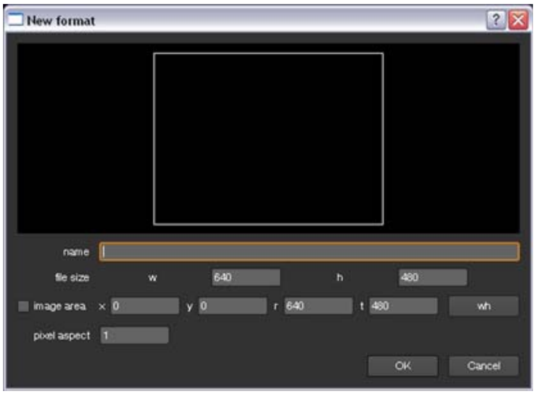
Note *The full-size format does not affect the format of the elements you read into your script. Nuke is resolution-independent, which means it will respect and keep the resolution of your images. It won't automatically crop or pad elements to match the project settings.*

If you want elements you read in to conform to the project settings, you can do this manually using the Reformat node. For more information, see "Reformatting Image Sequences" on page 139.

The full-size format is also used to calculate proxy scaling if a proxy format is used. For more information on the proxy mode and proxy formats, see "Proxy Mode" on page 124.

To set up a full-size format:

1. If it's not already open, select **Edit** > **Project Settings** (or press S) to display the Project settings dialog.
2. From the **full size format** pulldown menu, select the resolution for the final output of rendered images. If the format you want to use is not in the list, select **new**. The *New format* dialog appears.



file size w 640 h 480
☐ image area x 0 y 0 r 640 t 480
pixel aspect 1

In the **name** field, enter a name for the new format.

In the **file size** fields, define the width and height of the format.

If you like, you can also define additional information, such as offsets and pixel aspect ratio.

Click OK to save the format. It now appears in the pulldown menu where you can select it.

Proxy Mode

When compositing with Nuke, you can work in two different modes: the *full-size mode* or *proxy mode*. In the full-size mode, images are read in exactly as they are on the disk, and all positions are in actual pixels in these images. This is the mode you want to use for accurate feedback and when rendering the final output.

In the proxy mode, instead, a proxy scale factor is used. All images and all x/y positions are scaled by this factor. This will produce the same (or at least very similar) composite at a different scale. For example, you can use a fraction of the full output resolution to speed up rendering and display calculation.

In addition to the above, a separate proxy file can also be read in place of a full-size image, provided you have specified one in the relevant Read node. This can further speed up the preview, by using a smaller image that reads faster and also saves time by not needing to be scaled. For more information, see "Read nodes and proxy files" on page 126.

The proxy settings you define in the project settings affect both the proxies Nuke generates using the proxy scale factor and proxies read from files. Below, we discuss setting a proxy format and/or a proxy scale, and defining how Read nodes use proxy files.

Note *Note that proxy versions of images are only used if you have activated the proxy mode (see "Toggling in and out of Proxy mode" on page 128). When the proxy mode is off, Nuke will always use the full-res files.*

Proxy format and proxy scale

In the Project settings dialog, you have the option of defining a *proxy format* and/or a *proxy scale* that you use in the proxy mode.

For the proxy *format*, you can define the image resolution as well as additional information about offsets and pixel aspect ratio. When using the proxy format, the scaling is proportionate to the full-size/proxy format relationship (not scaled to the proxy format).

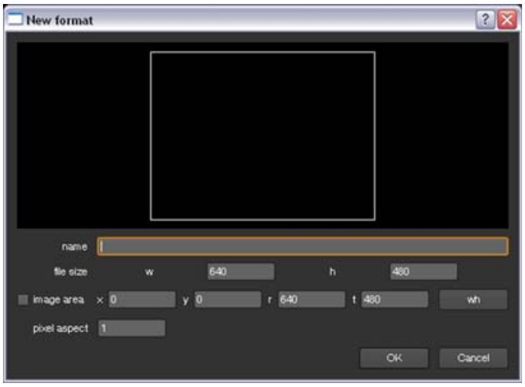
For the proxy *scale*, you only define a simple scale factor by which your images are scaled down whenever the proxy mode is activated. For example, you can use the scale factor of 0.5 to scale your images down to half the size.

If you like, you can define both a proxy format and a proxy scale, and then choose which one to use in proxy mode. A proxy scale is easier to set up, but a proxy format gives you more control over the low-res versions of your images. Below, we first describe how to set up proxy formats and then how to define a proxy scale.

To set up proxy formats:

1. If it's not already open, select **Edit** > **Project Settings** (or press S) to display the Project settings properties panel.
2. If you want to use a proxy format (rather than a proxy scale) whenever the proxy mode is activated, select **format** from the pulldown menu on the right.
3. From the **proxy format** menu, select the resolution to use while working to speed things up. Notice that your images are not scaled to this resolution, but the scaling is proportionate to the full-size/proxy format relationship. Nuke divides the proxy format width by the full-size width and uses the result as the scale factor.

   If the proxy format you want to use is not in the list, select **new**.The *New format* dialog appears.



file size w 640 h 480

☐ image area x 0 y 0 r 640 t 480 wh

pixel aspect 1

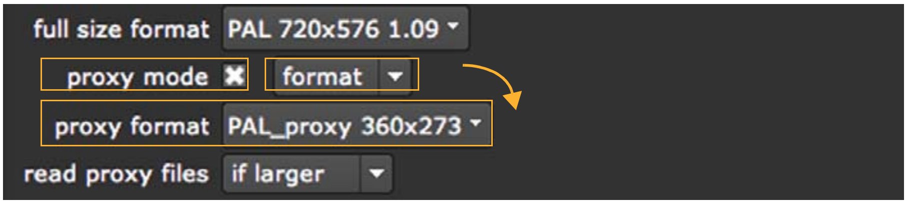
   In the **name** field, enter a name for the new format.

   In the **file size** fields, define the width and height of the format.

Tip *You can type formulas in numeric fields to do quick calculations. For example, if your full-size format width is 4096 and you want your proxy format width to be 1/2 of that, you can enter **4096/2** in the **New format** dialog's **file size w** field and press **Enter**. Nuke then calculates the new width for you.*

   Click OK to save the format. It now appears in the pulldown menu where you can select it.

4. To activate the proxy mode and use the low-res format for calculations and display, check **proxy mode**.

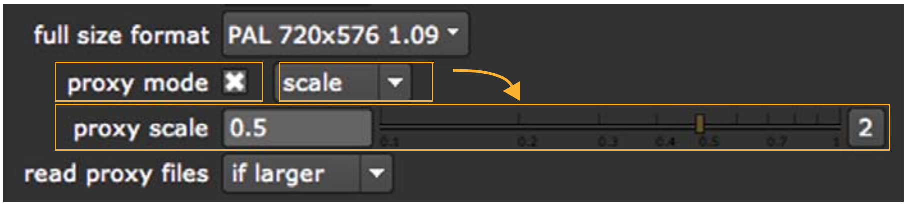


full size format PAL 720x576 1.09
proxy mode ☒ format
proxy format PAL_proxy 360x273
read proxy files if larger

Alternatively, you can use the proxy toggle in the Viewer controls, or press **Ctrl+P** (**Cmd+P** on a Mac). For more information, see "Toggling in and out of Proxy mode" on page 128.

To setup a proxy scale:

1. If it's not already open, select **Edit** > **Project Settings** (or press **S**) to display the Project Settings properties panel.
2. Select **scale** from the pulldown menu in the Project Settings dialog.
3. Using the **proxy scale** input field or slider, specify the factor by which you want to scale the width and height of your images. For example, if you want to scale them down by 50%, use the value of **0.5**.
4. To activate the proxy mode and use the low-res format for calculations and display, check **proxy mode**.



full size format PAL 720x576 1.09
proxy mode ☒ scale
proxy scale 0.5
read proxy files if larger

Alternatively, you can use the proxy toggle in the Viewer controls, or press **Ctrl+P** (**Cmd+P** on a Mac). For more information, see "Toggling in and out of Proxy mode" on page 128.

Read nodes and proxy files

As an alternative to letting Nuke generate proxies on the fly, proxy files can be specified using a second filename in the Read nodes (for how to do this, see "Loading Image Sequences" on page 134). If you don't have a proxy file, you can create one by activating the proxy mode and rendering your full-size images using a Write node (see "Rendering Output" on page 522).

The proxy file does not need to match the proxy resolution in use. Depending on your project settings, either the full-res or proxy file will be scaled to the required proxy size (that is, the size calculated by taking the full-size format and scaling it by the current proxy settings). However, if your proxy images match your typical proxy settings, you can save this time.

To define which file (full-res or proxy) is used in the proxy mode:

1. If it's not already open, select **Edit** > **Project Settings** (or press S) to display the Project Settings properties panel.
2. From **read proxy files**, select when to use the proxy file (rather than the full-res file) in a Read node:
    - **never** - Never use the proxy file in the proxy mode. Instead, scale the full-size file as necessary.
    - **if larger** - Use the smaller of the two images if it is larger or equal to the desired size, scaling down as needed. Otherwise, use the larger one, scaling down or up as needed. This is the default option.
    - **if nearest** - Use the image that is closest to the desired size, scaling up or down as needed.
    - **always** - Always use the proxy image in the proxy mode, scaling it up or down as necessary.

    The option you choose affects all Read nodes in your script, provided that a proxy file is named and the proxy mode is on.

Write nodes and proxy files

It is worth mentioning here that when a script is rendered in proxy mode, processing will be done at the proxy scale and image output will go to the file name in the Write node's **proxy** field. If you do not specify a proxy file name, the render will fail with an error. It never resizes the proxy image, and it will not write the proxy image over the full-size one.

For more information, see "Rendering Output" on page 522.

Using Proxy mode to enlarge a script

If you find it necessary to render a comp at higher than the original resolution it was intended for, you can also use the proxy resolution to scale up from the root full size format. For example, you could use a 2K composite to produce 4K or higher images in the proxy mode.

The reason you'd probably want to do this is because the comp you did needs to be re-run at higher resolution for a different output target and possibly with some new higher resolution elements. For example, you may need to re-render with new CG elements at print resolution rather than for film out, but you don't want to go through your script and modify everything that has an x/y position.

When scaling up the output of a script in the proxy mode, image generator nodes will render at the larger size, larger images can be specified in the proxy field, and as with scaling down, all x/y positions will be scaled to the proxy output resolution.

Using small proxy files

If you have previously set up your script to use small proxy files, you do not have to remove these. Make sure **read proxy files** in Project Settings is set to anything other than **always**, and Nuke will read the larger original files and scale them up.

Using large proxy files

If you actually have larger proxy files, you should enter them into the Read nodes' **proxy** field, and set **read proxy files** to anything other than **never**. Nuke will then use these larger files in the proxy mode. For maximum quality, these should be at exactly the desired proxy size so that no scaling is done to them. For example, if the required proxy size (defined by the project settings) is 4K, then the proxy image should be exactly 4K. Otherwise, Nuke will scale it to match the project settings, which will reduce the quality.

Rendering the scaled-up output

To render the larger output of a scaled-up script, you need to activate the proxy mode and enter a file name in the **proxy** field of the Write nodes. The larger images will then be written to these files.

Toggling in and out of Proxy mode

It's usually smart to work in proxy mode because most operations work quickly and more efficiently under the low-res display. You can switch between low- and high-resolution when you need greater precision (for example, when pulling a key or tracking), or when you're ready for final rendering.

To toggle between full resolution and Proxy mode:

1. Click on an empty area of the Nuke window.
2. Press **Ctrl+P** to toggle the display mode (**Cmd+P** on a Mac).

Nuke automatically scales script elements—Bezier shapes, B-spline shapes, paint curves, garbage masks, tracking curves, and so on—to keep the original placement on the image.

Image Caching

To ensure fast playback, the Nuke Viewer saves a version of every frame it displays to disk. When you play through sequences in the Viewer, it reads, where possible, from this cache of prerendered images. This way, Nuke can quickly display the results without recalculating the parts of the script that have not changed. Provided the caching preferences (see "Defining the Settings for Caching" on page 129) have been set, Nuke automatically caches any output displayed in a Viewer. However, you can also use the DiskCache node (see "Using the DiskCache Node" on page 130) to cache other parts of the node tree.

The Cache Directory

Both the automatic caching and the DiskCache node use the same cache directory defined using the Preferences dialog. In the Preferences, you can also set the maximum size you allow the disk cache to consume. For more information on the caching preferences, see "Defining the Settings for Caching" below.

Note that the cached images have unique names reflecting their point of output location in the script. This means that you can cache images from multiple nodes in the script without overwriting previously cached images.

Defining the Settings for Caching

You can define the settings for both the automatic caching and the DiskCache node in the Preferences.

To define the settings for caching:

1. Select **Edit** > **Preferences** (or press Shift+S).

   The Preferences dialog opens.
2. Under **disk cache**, specify where you want Nuke to cache out data to disk. Pick a local disk (for example, c:/temp), preferably with the fastest access time available.

   The environment variable NUKE_DISK_CACHE can be used to override this setting. For more information, see "Environment Variables" on page 611.
3. Using the **disk cache size** control, pick the maximum size the disk cache can reach. Ensure there is enough space on the disk for this to be reached. The default value is 10GB, but a larger value, such as 50GB, can often be used.

The environment variable NUKE_DISK_CACHE_GB can be used to override this setting. For more information, see "Environment Variables" on page 611.

4. Click the **Save Prefs** button to update preferences and then restart Nuke.

Once these settings are defined, the automatic caching of images is enabled. The Viewer will cache each frame it displays in the directory specified. If you add a DiskCache node into your script, it will also use the same directory.

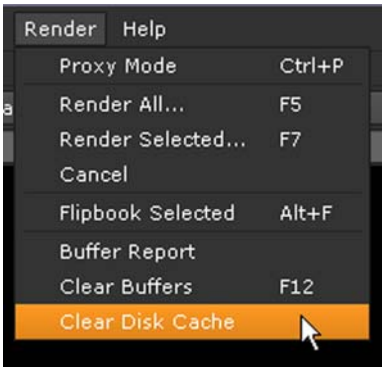
The disk cache is preserved when you quit Nuke so it can be used again later. However, when the cache becomes full, old items are automatically deleted to make room for newer items.

Clearing the Disk Cache

When the disk cache becomes full, old items are automatically deleted. If necessary, you can also empty the disk cache manually. You may want to do this if, for some reason, wrong images are displayed in the Viewer.

To empty the disk cache:

From the menu bar, select **Render** > **Clear Disk Cache**.



Using the DiskCache Node

The DiskCache node caches to disk scanlines from its input as they are requested by its output. This can be useful, for example, if:

- you are working on a large, complex node tree. Using the DiskCache node, you can break the node tree into smaller sections and cache any branches that you are no longer working on.
- you are reading in images from a network. If you insert a DiskCache node after a Read node, the image will be cached locally and displayed faster.

- you are painting or rotoscoping. If you insert a DiskCache node before a RotoPaint node, flipping frames becomes faster.

The cached images are saved in the same directory as the images the Nuke Viewer caches automatically. You can set the location and size of this directory in the Preferences. For more information, see "Defining the Settings for Caching" on page 129.

Note *Even though the DiskCache node and the automatic Viewer caching use the same cache directory, they do not share the same cache files. Therefore, using a DiskCache node does not create cache files for the Viewer and does not necessarily speed up playback. Instead, if placed at strategic, expensive parts of a node tree, it can speed up calculations, as Nuke can reference the cached data rather than recalculate it.*

Unlike the images in the Viewer cache, the images created by the DiskCache node affect your rendered output and are always saved as full floating point images.

If you make a change in the nodes upstream, the affected cached images are discarded and automatically recalculated.

Note *When executing a script on the command line, the DiskCache nodes are NOT automatically executed.*

To cache images upstream:

1. Set the zoom level in the Viewer. By default, only the lines displayed in the Viewer will be cached.
2. Select **Other** > **DiskCache** to insert a DiskCache node after the last node in the section of the node tree that you want to cache.
3. From the **channels** menu, select which channels to cache.

   Nuke caches the selected channels of the current frame at the current zoom level. From this point on, Nuke references the cached data instead of constantly recalculating the output of the preceding nodes.

   As you pan and zoom around, new parts of the image are cached.
4. If you want to cache more than the current frame and zoom level, click the **Precache** button in the DiskCache properties and enter a frame range in the dialog that opens.

   This forces Nuke to cache all frames specified. All lines will be cached regardless of what is shown in the Viewer. Where the required images are partly cached already, Nuke only calculates what is missing.

Saving Scripts and Recovering Backups

You know the mantra: save and save often. Nuke provides three ways to save your scripts, plus an automatic timed backup.

Saving Scripts

There are three ways of saving scripts:

- To save a new script, select **File** > **Save as** (or press Shift+Ctrl/Cmd+S).
- To update changes to a script already saved, **File** > **Save** (or press Ctrl/Cmd+S).
- To save and upgrade to the next version, **File** > **Save new version** (or press Alt+Shift+S).

To save a script:

1. Select **File** > **Save as**.

   The *Save script as* dialog opens.
2. Browse to the directory where you want to store the script. For instructions on using the file browser, see "Using the File Browser" on page 111.
3. In the field in the bottom of the dialog, enter a name for the script after the folder path, for example **firstscript_v01.nk**.
4. Click **Save**.

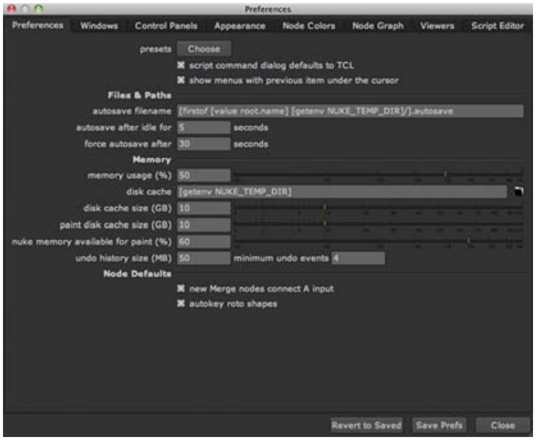
Tip *The **_v01** string in the end of a script name allows you to use the **Save new version** feature. Selecting **File** > **Save new version** saves the current version of your script and increments its name (that is, saves the different versions under different names using _v01, _v02, _v03, and so on, in the end of file names). This only works when the file name includes a number that can be incremented.*

Automatic Backup of Scripts

You can define where and how often Nuke makes automatic backups your files, or turn off the autosave function.

To define autosave options for a script:

1. Select **Edit > Preferences**.
   The Preferences dialog opens.



presets

☒ script command dialog defaults to TCL

☒ show menus with previous item under the cursor

Files & Paths

autosave filename [firstof [value root.name] [getenv NUKE_TEMP_DIR]/].autosave

autosave after idle for 5 seconds

force autosave after 30 seconds

Memory

memory usage (%) 50

disk cache [getenv NUKE_TEMP_DIR]

disk cache size (GB) 10

paint disk cache size (GB) 10

nuke memory available for paint (%) 60

undo history size (MB) 50 minimum undo events 4

Node Defaults

☒ new Merge nodes connect A input

☒ autokey roto shapes

2. Edit the following settings:
   - **autosave filename** to define where and under what name Nuke saves your automatic backup files. By default, the files are saved in the same folder as your project files with the extension **.autosave**. To change this, enter a full directory pathname in the **autosave filename** field.
   - **autosave after idle for** to define how long (in seconds) Nuke waits before performing an automatic backup after you have left the system idle.
   - **force autosave after** to define how long (in seconds) Nuke waits before performing an automatic backup regardless of whether the system is idle.
3. Click **Save**.

Note *If you close the Preferences dialog without clicking **Save Prefs**, your changes will only affect the current session of Nuke.*

Note *For the automatic backup to work, you must save your script first so that the autosave can reference the file. We'd hate for you to lose your work, so please do this early on in the process!*

To turn off automatic backup:

1. Select **Edit > Preferences**.

   The Preferences dialog opens.
2. Set the **autosave after idle for** and **force autosave after** fields to 0.

From now on, Nuke will not perform any automatic backups, and you are more likely to lose your work in the case of a system or power failure.

Recovering Backups

After experiencing a system or power failure, you are likely to want to recover the backup files created by Nuke's autosave function.

To recover backups:

1. Relaunch Nuke.

   A dialog opens that asks you if you want to recover the autosave file.

2. Click **OK**.

Nuke opens the backup file for your use.

There may be times when you don't want to load the autosave file and rather need to load the last saved version. For example, consider a situation where you modified a script, but decided not to commit the changes and so exited Nuke without saving. In all likelihood Nuke autosaved some or all of your changes, in which case if you open the autosaved file you will not be working on the original script, as intended. If you accidentally open an autosaved script, then simply close it and reload the last saved version.

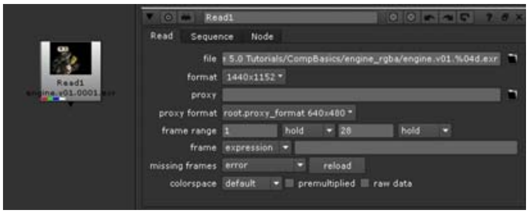
Loading Image Sequences

When you are ready to start compositing, you may want to begin by importing a background or foreground image sequence. Typically, you would read in both full- and proxy-resolution versions of the sequence. You can read in several image sequences in one go.

To import an image sequence:

1. Select **Image** > **Read** (or press **R** over the Nuke window).

   A Read node is inserted in the Node Graph.



file ː 5.0 Tutorials/CompBasics/engine_rgba/engine.v01.%04d.exr
format 1440x1152
proxy
proxy format root.proxy_format 640x480
frame range 1 hold 28 hold
frame expression
missing frames error
colorspace default ☐ premultiplied ☐ raw data

2. Browse to the image sequence you want to import.

   For instructions on using the file browser, see "Using the File Browser" on page 111. Select the file you want to open. If you want to open several files at the same time, **Ctrl+click** (Mac users **Cmd+click**) on the files. Click **Open**.

   Nuke imports the image sequence and displays it as a thumbnail on the Read node. Generally, the Read node does not reformat or resize the sequence in any way, and the node's properties panel is updated to display the native resolution and the frame range for the sequence. Note that the **format** and **proxy format** fields in the controls indicate the format of the images, they do not cause the images read from files to be resized to this format.

3. If your sequence has a red, green and blue channel but no alpha channel, Nuke steps in and automatically sets the alpha channel to 1. This prevents possible problems from occurring if Nuke tries to read the alpha channel and one doesn't exist. You can uncheck the **auto alpha** box in the Read node control panel if you want to cancel the alpha being added.

Note *Nuke reads images from their native format, but the Read node outputs the result using a linear colorspace. If necessary, you can change the Colorspace option in the Read node's properties panel, or insert a **Color > Colorspace** node to select the color scheme you want to output or calculate.*

Note *The maximum image size the Nuke Viewer can display is 2^28 = 268,435,456 pixels. This is the same as 16k x 16k, 32k x 8k, 64k x 4k, or 128k x 2k. If your image is larger than this, it will be resized and you will get the following warning:*

"Viewer image is clipped to <size> x <size>!"

For example, if your image resolution is 60,000 x 4473, Nuke is able to display the image because the number of pixels is less than 2^28. However, if the resolution is 60,000 x 4474 (more than 2^28 pixels), the image will be resized to 59998 x 4474.

In addition to the Viewer, this limit is also applied to the bounding box of the images being passed between each node.

4. If you have a proxy version of the image sequence, click the proxy field's folder icon and navigate to the proxy version. Select **Open**.
   If you don't have a proxy version, don't worry: Nuke will create one on the fly according to the proxy scale or proxy format settings you specified in the project settings.

   The proxy file does not need to match the proxy resolution in use. Depending on your settings, either the full-res or proxy file will be scaled to the required proxy size. For more information, see "Read nodes and proxy files" on page 126.

Note *QuickTime .mov files may appear different in Nuke relative to Apple's Final Cut Pro, because Final Cut Pro introduces a gamma compensation based on assumptions about the content of the files and the viewing environment.*

Note *QuickTime is only supported by default on 32-bit Windows and Mac OS X (32-bit and 64-bit).*

Note *The EXR file format allows the display window to have the lower left corner in any position. Unfortunately, Nuke needs all formats to have a lower left corner coordinate of 0,0. In the Read node control panel, under* ***exr Options****, you can check the* ***offset negative display window*** *box to tell Nuke to offset the image so that the display window left side x coordinate is 0. If you uncheck the box, Nuke will shrink the format from both sides the amount that it's negative in the x coordinate, in other words treat the area as overscan.*

Note *On Linux, the Nuke EXR reader uses a memory-mapping function to improve performance reading PIZ-compressed EXR files. However, some customers have experienced hanging when reading large (frame size and number of channels) PIZ-compressed EXR files across an NFS network. If you experience this problem, you can tell Nuke not to use the mmap function by enabling this option. You can set it on a case by case basis or use a knobDefault in your init.py to always have it disabled. For more information on knobDefault, see "Setting Default Values for Controls" on page 571.*

Loading Images from an External File Browser

To load an image, you can also drag and drop the image into the Node Graph from an external file browser (such as Windows Explorer or Mac OS X Finder). To load an entire image sequence this way, drag and drop the directory that contains the images into the Node Graph.

Naming Conventions

The file names of image sequences generally end in a number before the extension, for example image0001.rgb, image0002.rgb, image0003.rgb, and so on. When browsing for files like this, you may notice that the sequence appears as **image####.rgb**. Here, **####** is Nuke's way of indicating that the number is in a 4-digit incremental format. For a 3-digit format, such as image001.rgb, the frame number variable would be **###**.

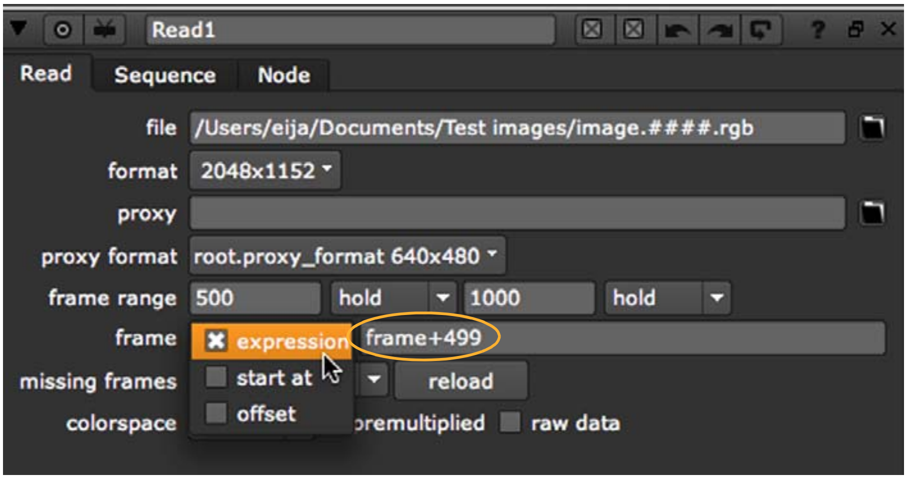
Nuke's File Browser also understands unpadded file names, such as **image1.rgb**, **image2.rgb**, **image3.rgb**, and so on. They appear as **image#.rgb**.

Changing the Relation Between the Current Frame and the Frame Read In

By default, Nuke assumes an exact relation between the current frame processed, and the frame read in. For example, at frame 15, Nuke reads in image.0015.rgb. However, you can change this behavior using the **frame** parameter on the Read node. For instance, if you have a sequence that runs from image.0500.rgb to image.1000.rgb, you may want to read in image.0500.rgb at frame 1. Nuke lets you do this via expressions, specified start frames, and constant offsets. Each method is described below.

Using expressions

1. Select **Image** > **Read** to import an image sequence.
2. In the Read node controls, set the **frame** pulldown menu to **expression**. Enter an expression in the field on the right. The expression changes the relation between the current frame and the frame read in.



Read1

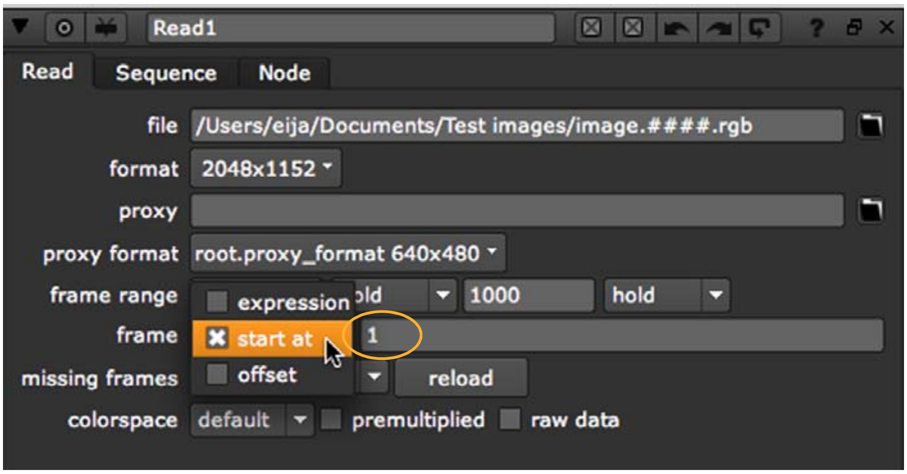
file /Users/eija/Documents/Test images/image.####.rgb
format 2048x1152
proxy
proxy format root.proxy_format 640x480
frame range 500 hold 1000 hold
frame ☒ expression frame+499
missing frames
colorspace

For example, if your clip begins from image.0500.rgb and you want to place this first frame at frame 1 rather than frame 500, you can use the expression **frame+499**. This way, 499 frames are added to the current frame to get the number of the frame that's read in. At frame 1, image.0500.rgb is read in; at frame 2, image.0501.rgb is read in; and so on.

Another example of an expression is **frame*2**. This expression multiplies the current frame by two to get the number of the frame that's read in. This way, only every other frame in the clip is used. At frame 1, image.0002.rgb is read in; at frame 2, image.0004.rgb is read in; at frame 3, image.0006.rgb is read in; and so on.

Specifying a start frame for a clip

1. Select **Image** > **Read** to import an image sequence.
2. In the Read node controls, set the **frame** pulldown menu to **start at**. Enter a start frame number in the field on the right. This specifies the frame where the first frame in the sequence is read in. In other words, all frames are offset so that the clip starts at the specified frame.

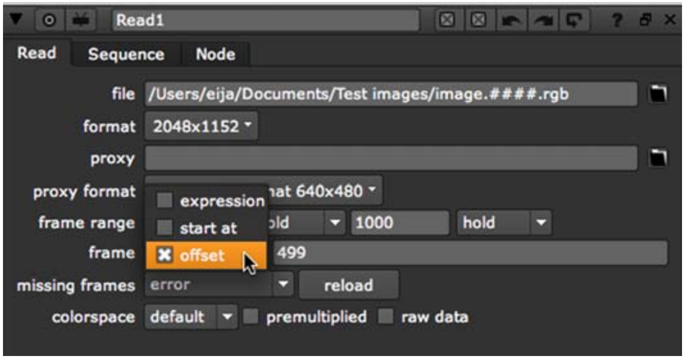


file /Users/eija/Documents/Test images/image.####.rgb
format 2048x1152
proxy
proxy format root.proxy_format 640x480
frame range 1000 hold
frame ☒ start at 1
missing frames
colorspace default ☐ premultiplied ☐ raw data

For example, if your sequence begins from image.0500.rgb and you enter **1** in the field, image0500.rgb is read in at frame 1. Similarly, if you enter **100** in the field, image0500.rgb is read in at frame 100.

Offsetting all frames by a constant value

1. Select **Image** > **Read** to import an image sequence.
2. In the Read node controls, set the **frame** pulldown menu to **offset**. Enter a constant offset in the field on the right. This constant value is added to the current frame to get the number of the frame that's read in.



file /Users/eija/Documents/Test images/image.####.rgb
format 2048x1152
proxy
proxy format 640x480
frame range 1000 hold
frame ☒ offset 499
missing frames error
colorspace default ☐ premultiplied ☐ raw data

For example, if your clip begins from image.0500.rgb and you want to place this first frame at frame 1 rather than frame 500, you can use **499** as the constant offset. This way, 499 is added to the current frame to get the frame that's read in. At frame 1, image.0500.rgb is read in; at frame 2, image.0501 is read in, and so on.

You can also use negative values as the constant offset. For example, if you use the value -10, Nuke will subtract ten from the current frame to get the frame that's read in. At frame 20, image.0010.rgb is read in; at frame 21, image.0011.rgb is read in; and so on.

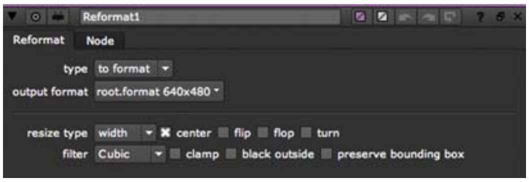
Reformatting Image Sequences

When you import image sequences, Nuke stores their format settings and makes them available to the Reformat node. You can then use the Reformat node to resize and reposition your image sequences to a different format. Reformat nodes also allow you to use plates of varying image resolution on a single script without running into issues when combining them.

To insert a Reformat node:

1. Make sure the Read node you added is currently selected.
2. Select **Transform** > **Reformat**.

   The Reformat node is inserted in the script, and its properties panel opens.



type: to format
output format: root.format 640x480
resize type: width ☒ center ☐ flip ☐ flop ☐ turn
filter: Cubic ☐ clamp ☐ black outside ☐ preserve bounding box

3. From the **output format** pulldown menu, select the format to which you want to output the sequence. If the format does not yet exist, you can select **new** to create a new format from scratch. The default setting, **[root.format],** resizes the image to the format indicated on the Project Settings dialog box.
4. You can now use the same Reformat node for any other Read nodes in the script. Simply select the Reformat node and **Edit** > **Copy**. Select another Read node in the script and **Edit** > **Paste**.

Loading Nuke Scripts

When you have built a script and saved it and want to come back to it later, you need to load in an entire script file. You recognize Nuke's script files from the extension **.nk** (for example **firstscript.nk**).

To load a script:

1. Select **File** > **Open** (or press **Ctrl/Cmd+O**).

   The *Script to open* dialog appears.

2. Browse to the script you want to open. For instructions on using the file browser, see "Using the File Browser" on page 111.
3. Click **Open**.

Closing Nuke Scripts

To close a script:

1. Select **File** > **Close** (or press **Ctrl/Cmd+W**).
2. If you have made any unsaved changes to the script, Nuke prompts you to select whether to save them. Click **Yes** to save your changes or **No** to ignore them.

   Nuke is quit and relaunched, as though you ran it again. It does everything it does at start-up, apart from displaying the splash screen. Therefore, you can use **Ctrl/Cmd+W** as a quick way to clear memory, reread plug-in paths, and reload the init.py and menu.py files. (The init.py and menu.py files are files that Nuke runs at start-up and can be used for configuring Nuke. You may want to reload these files if you have made any changes to them.)

Defining Frame Ranges

Several dialogs in Nuke, such as the *Frames to render* and *Frames to flipbook* dialogs, prompt you for a frame range. To define one, you need to enter a starting frame and an ending frame, separated by a dash. For example, to restrict an action to frames 1, 2, 3, 4, and 5, you would use **1-5** as the frame range.

The following table gives you more examples of frame ranges you can define.

Frame Range	Expands To
3	3
-3	-3
1 3 4 8	1, 3, 4, 8
1-10	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
-3-4	-3, -2, -1, 0, 1, 2, 3, 4
-8--5	-8, -7, -6, -5
1-10x2 (frame range from 1 to 10 in steps of 2)	1, 3, 5, 7, 9

Frame Range	Expands To
1-10x3 (frame range from 1 to 10 in steps of 3)	1, 4, 7, 10
1-4x1 8-10x1 12-14x1 (multiple ranges separated by spaces)	1, 2, 3, 4, 8, 9, 10, 12, 13, 14

You can use the above ways of defining a frame range everywhere in Nuke. In addition to dialogs, they can be used on the command line (where any frame ranges should be preceded by the -F switch) and in Python statements. For more information, see "Command Line Operations" on page 604 and "Setting Frame Ranges Using Python" on page 574.

File Name Search and Replace

With the Search and Replace function, you can quickly replace all or part of file names or file paths in any node with **file** or **proxy** controls (for example in Read and Write nodes).

To search for a file name or file path and replace it:

1. Select the node(s) where you want to replace all or part of a file name or file path.
2. Select **Edit** > **Node** > **Filename** > **Search and Replace**.

   OR

   Press Ctrl+Shift+/ (Mac users press Cmd+Shift+/).
3. In the dialog that opens, enter the string you want to search for and the string you want to replace it with. Click **OK**.

Nuke searches for the string in the selected nodes and replaces it with the new string.

Note *You can also enter expressions into the Search and Replace dialog. Just remember that the search field in the dialog only takes regular expressions. Any characters that have specific meanings in regular expressions, such as **[** and **]**, need to be preceded by the \ character. For example, **[getenv HOME]** would need to be entered as **\\[getenv HOME\\]**.*

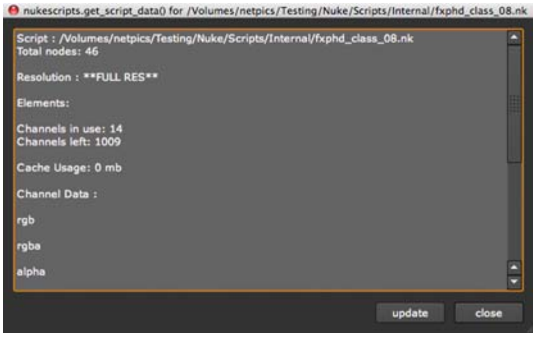
*You can also pass flags alongside the expression itself to control how the expression behaves. For example, to perform case-insensitive searches, you can enter **(?i)** in the beginning of the expression or after one or more whitespace characters.*

Displaying Script Information

To display script information, such as the node count, channel count, cache usage, and whether the script is in full-resolution or proxy mode, do the following:

1. Select **File** > **Script Info** (or press Alt+I).

   The script information window opens.


nukescripts.get_script_data() for /Volumes/netpics/Testing/Nuke/Scripts/Internal/fxphd_class_08.nk
update
close

```
Script : /Volumes/netpics/Testing/Nuke/Scripts/Internal/fxphd_class_08.nk
Total nodes: 46

Resolution : **FULL RES**

Elements:

Channels in use: 14
Channels left: 1009

Cache Usage: 0 mb

Channel Data :

rgb

rgba

alpha
```

2. If you make changes to your script while the window is open, click **update** to update the information.
3. To close the information window, click **close**.

Grouping Nodes in the Node Graph

You can group nodes in the Node Graph using the Backdrop node or the Group node. The Backdrop node adds a background box behind the nodes, separating the nodes visually from the rest of the node tree. A Group node, instead, combines a set of nodes into a single node, acting as a nesting container for those nodes.

Grouping Nodes with the Backdrop Node

You can use the Backdrop node to visually group nodes in the Node Graph. Inserting a Backdrop node creates a box behind the nodes. When you move the box, all the nodes that overlap the box are moved, too. By inserting several backdrop nodes, you can group the nodes in your node tree onto boxes of different colors and titles. This makes it easier to find a particular node in a large node tree, for example.

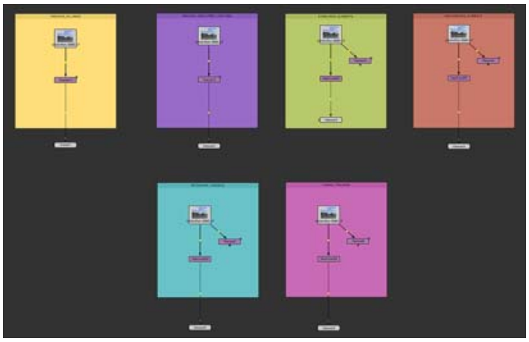
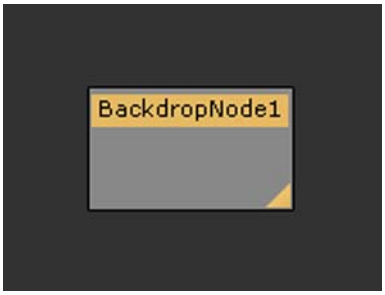


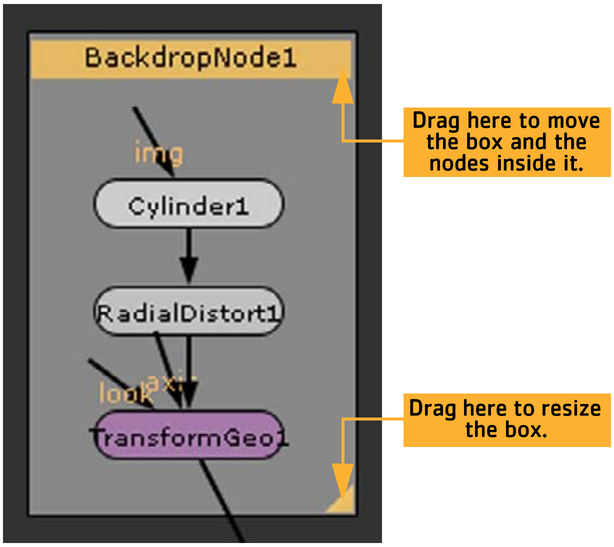
Figure 4.1: Nodes grouped in the Node Graph with Backdrop nodes.

To group nodes with a Backdrop node:

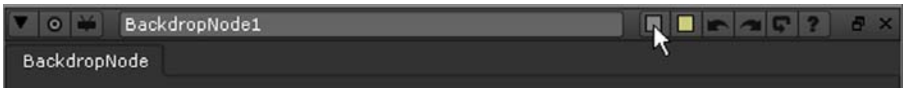
1. Select **Other** > **Backdrop**. A Backdrop node box appears in the Node Graph.



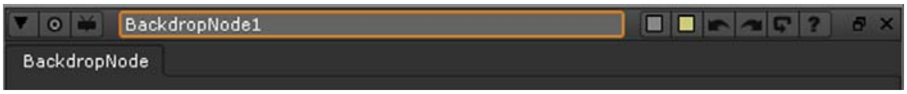
2. Drag the triangle in the lower right corner of the box to resize the box as necessary.
3. Click on the box title bar and drag it to move the box behind the nodes you want to group together. If there are any nodes on the box, they move together with the box.


BackdropNode1
img
Cylinder1
RadialDistort1
look
axis
TransformGeo1
Drag here to move the box and the nodes inside it.
Drag here to resize the box.

4. To change the color of the box, open the Backdrop node's controls by double-clicking on the title bar, then click the left color button and pick a new color with the color picker that appears.


BackdropNode1
BackdropNode

5. To change the title of the box, enter a new name for the Backdrop node in the node's controls.


BackdropNode1
BackdropNode

6. If you later want to remove both the box and the nodes inside it, click on the title bar and press **Delete**. **Ctrl/Cmd**+clicking on the title bar and pressing **Delete** only removes the box and leaves the nodes untouched. To remove the nodes and leave the box untouched, click on the triangle in the lower right corner of the box and press **Delete**.

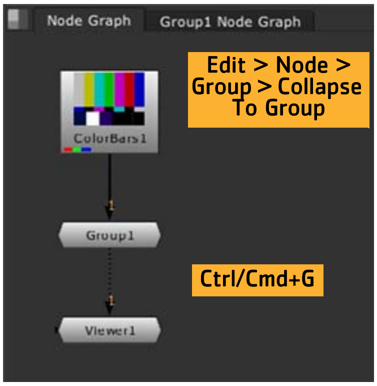
Grouping Nodes with the Group Node

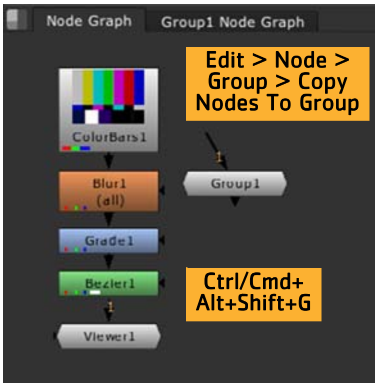
You can use the Group node to nest multiple nodes inside a single node.

To group nodes with a Group node:

1. Select all the nodes you want to nest inside the Group node.
2. If you want to replace the original nodes with the Group node, select **Edit** > **Node** > **Group** > **Collapse To Group** (or press Ctrl/Cmd+G on the Node Graph).

   If you want to keep the original nodes in the layout in addition to the Group node, select **Edit** > **Node** > **Group** > **Copy Nodes To Group** (or press Ctrl/Cmd+Alt+Shift+G on the Node Graph).


Node Graph
Group1 Node Graph
Edit > Node > Group > Collapse To Group
ColorBars1
Group1
Ctrl/Cmd+G
Viewer1

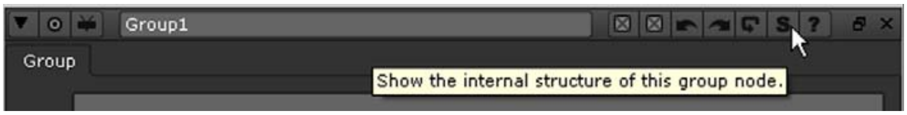

Node Graph
Group1 Node Graph
Edit > Node > Group > Copy Nodes To Group
ColorBars1
Blur1 (all)
Group1
Grade1
Bezier1
Ctrl/Cmd+ Alt+Shift+G
Viewer1

The selected nodes are nested into a group. The internal structure of the Group node is shown on a separate tab that opens.

Tip *As an alternative to* ***Edit*** *>* ***Node*** *>* ***Group*** *>* ***Collapse to Group****, you can also select* ***Other*** *>* ***Group*** *from the Toolbar or the Node Graph right-click menu.*

To view the nodes nested inside a Group node:

In the Group node's controls, click the **S** button (short for Show) in the top right corner.


Group1
S
?
Group

Show the internal structure of this group node.

A new tab that contains the nested nodes opens.

To ungroup nodes:

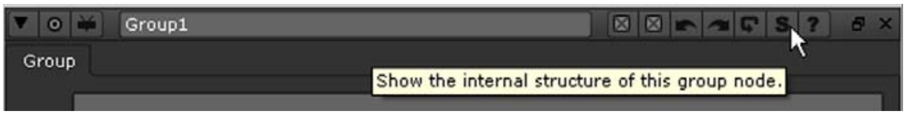
1. Select the Group node in the Node Graph.

2. Select **Edit** > **Node** > **Group** > **Expand Group** (or press **Ctrl/Cmd+Alt+G**).

   The Group node is replaced with the nodes that were nested inside it.

OR

1. In the Group node's controls, click the **S** button in the top right corner.



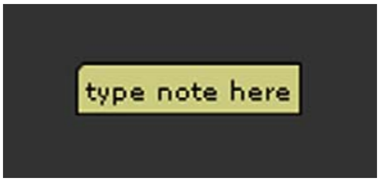
   A new tab that contains the nested nodes opens.
2. Copy the nodes from the new tab into your script. If you want to lock the connections between the grouped nodes so that they cannot be accidentally disconnected during the copy-paste operation, check **lock all connections** in the Group node's controls.
3. Delete the unnecessary Group node from your script.

Adding Notes to the Node Graph

Using the StickyNote node, you can add notes to the Node Graph. The notes can be any text or HTML mark-up. Usually, they are made as annotations to the elements in the node tree.

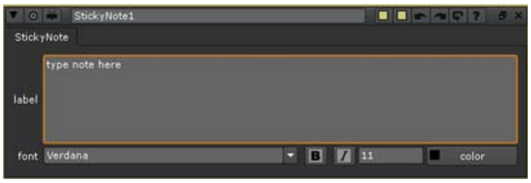
To add a note to the Node Graph:

1. Click on the part of the Node Graph where you want to add a note.
2. Select **Other** > **StickyNote**. A note box appears in the Node Graph.



3. In the StickyNote controls, enter your note in the **label** field. If you like, you can use HTML mark-up. For example,
   - to have a note appear in bold, you can use **my note**. This would appear as **my note**.
   - to have a note appear in italics, you can use **<i>my note</i>**. This would appear as *my note*.
   - to add an icon to your note, you can use ****. This adds the Nuke color wheel icon. You can also use your own icons in the same way as long as you save them in your plug-in path direc-

tory. Most common image formats will work, but we recommend using PNG files.



label: type note here
font: Verdana 11

Using the Precomp Node

The Precomp node is like a Group node, but its content is stored in an independent .nk file. This allows you to save a subset of the node tree as a separate .nk script, render the output of this saved script, and read the rendered output back into the main comp as a single image input.

Precomp nodes can be useful in at least two ways. Firstly, they can be used to reduce portions of the node tree to pre-rendered image inputs. This speeds up render time, as Nuke only has to process the single image input instead of all the nodes that were used to create it. Because the original nodes are saved in a separate .nk script, you also maintain access to them and can adjust them later if necessary.

Secondly, Precomp nodes enable a collaborative workflow. While one artist works on the main comp, others can work on the sections that have been exported using the Precomp node. These sections can be edited, versioned, and managed independent of the main comp. For example, say you have a comp that involves a complex, multi-layered CG render. A 3D artist can produce this as a separate script that the compositor finishing the shot then reads in using a Precomp node. This way, the 3D artist can modify and re-render the CG element portion of the comp without having to share the main comp with the compositor.

Creating Precomp Nodes

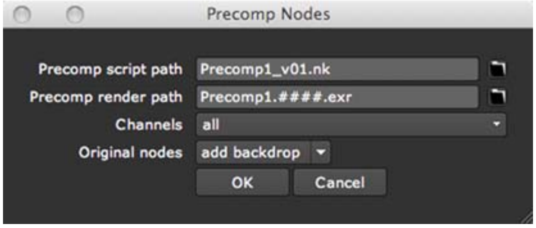
Whether you want to use the Precomp node to speed up rendering or to enable a collaborative workflow, the process of creating the Precomp is the same.

To create a Precomp node:

1. Select the nodes you want to include in the separate precomp script. If you select a Group node, the nodes nested inside it will be copied into the precomp script.

2. Select **Other** > **Precomp** (or press Ctrl/Cmd+Shift+P on the Node Graph).

   The *Precomp Nodes* dialog opens.



Precomp script path: Precomp1_v01.nk
Precomp render path: Precomp1.####.exr
Channels: all
Original nodes: add backdrop

Note *If you don't select any nodes when you create a Precomp node, the Precomp Nodes dialog is not displayed and the node is left blank. Using the* ***file*** *parameter in the Precomp controls, you can then browse to an existing script to load into the Precomp node.*

If you have several Write nodes in the existing script and want to control which of them is used for the output of the Precomp node, you can select ***Other*** *>* ***Output*** *to insert an Output node after the Write node you want to use.*

If the Precomp node cannot find an Output node in the script, it looks for a Write node and sets ***Output node*** *in the Precomp controls to the name of that node. The* ***Output node*** *field can also be used later to override what is set as the Output node in the precomped script. To do so, make sure you check* ***enable****. This check box allows you to toggle between the output node chosen by default and the output node you specified in the* ***Output node*** *field.*

3. Click the file browser icon next to **Precomp script path**, and browse to the directory where you want to save the precomp .nk script. After the directory path, enter a name for the precomp script, for example Precomp1_v01.nk. By default, the precomp script is saved next to the main script. If the main script has not been saved, the precomp script is saved in the current directory.
4. Click the file browser icon next to **Precomp render path**, and browse to the directory where you want to save the rendered output of the precomped nodes. After the directory path, enter a name for the rendered image, for example Precomp1_####.exr. If you like, you can also use version numbers in the name, for example Precomp1_v01_####.exr.

Important *We recommend rendering the image as an exr because that way Nuke will write the hash value of the incoming node tree into the rendered file. If the precomp script changes so that the hashes won't match, Nuke will then* *notify you, and you can update the resulting image. If you use a file format other than exr, you will not get this notification and the rendered file is likely to become out of date.*

For more information on hash values, see "What is the hash value?" on page 531.

5. From the **Channels** pulldown menu, select the channels you want to include in the rendered output of the precomped nodes.

   If you later need to adjust this selection, you can do so by setting the channels on the appropriate Write node (by default, Write1) in the precomp .nk script.
6. From the **Original nodes** pulldown menu, select:
   - **add backdrop** to create a backdrop behind the precomped nodes in the Node Graph.
   - **delete** to delete the precomped nodes.
   - **no change** to do nothing to the precomped nodes.
7. Click **OK**.

   Nuke saves the selected nodes in the .nk script specified. This script will also include input and output nodes, a Write node, and the current project settings.

   In the Node Graph, the selected nodes are replaced with the Precomp node. Nuke opens the properties of the Precomp node.

Precomp nodes and Project Settings

When you create a Precomp node, the precomp .nk script gets its project settings from the main script. The main script's project settings are also used whenever the precomp script is loaded into the main script. Therefore, if you open the precomp script in a separate instance of Nuke and change its project settings so that they no longer match the main script's settings, your changes do NOT have an effect when the precomp script is loaded into the main script. If you want to change the project settings, you should always do so in the main script rather than the precomp script.

Using a Precomp Node to Speed-up Rendering

If you want to use a Precomp node to speed up rendering, you need to have the Precomp node read in the output of the precomp script.

To render a Precomp node:

1. Create a Precomp node. For more information on how to do this, see "Creating Precomp Nodes" on page 147.

2. In the Precomp node controls, click **Render**. Enter a frame range (for example, 1-100) in the dialog that opens and click **OK**.

   Nuke renders the output of the precomp script and saves the resulting image in the **Precomp render path** directory you specified when you created the Precomp node.

   If you have activated the proxy mode and have specified a proxy filename in the output node, the rendered image is a proxy image.
3. If the render finishes successfully, **read file for output** is automatically turned on in the Precomp properties. When this is checked, the Precomp node turns green and reads in the rendered precomp image rather than calculates the output of the precomp script. The word **(Read)** and the name of the image read in are also displayed on the Precomp node in the Node Graph to indicate this.
4. In case there is an error opening the rendered output of the precomped nodes, select what to do from the **missing frames** menu in the Precomp properties:
   - **error** - display an error message on any missing frames.
   - **black** - replace any missing frames with black.
   - **checkerboard** - replace any missing frames with a checkerboard image.
   - **read input** - display the result of the input tree rather than the rendered file on any missing frames.

When several Precomp nodes are used like this to replace sections in a large, complex node tree, the render times may become significantly faster.

Note *When rendering the output of the precomp script, Nuke automatically selects the Write node to use by first looking for an Output node. If it can't find any Output nodes, it tries to find a Write node, and sets **Output node** in the Precomp node controls to the name of the Write node. If it can't find any Output or Write nodes, it produces an error.*

*At any point, the Precomp node properties can be used to override what is set as the Output node in the precomped script. To do so, open the Precomp properties and the **advanced** controls. In the **Output node** field, enter the name of the Write node whose output you'd like to use. Make sure you also check **enable**. This check box allows you to toggle between the output node chosen by default and the output node you specified in the **Output node** field.*

Tip *You can also use Write nodes in a similar manner and have their output read in from the rendered file rather than calculated using the nodes upstream.*

For more information, see "Using a Write Node to Read in the Rendered Image" on page 531.

Precomp Revisions

The following describes how to open, edit, and reload a precomp script.

To view and edit a precomp script:

1. In the Precomp node properties, click **Open**.

   This starts a new Nuke session and loads the precomp script.
2. Edit the precomp script as necessary.
3. If you are using version numbers in the Write node file name, select the Write node in the Node Graph and choose **Edit** > **Node** > **Filename** > **Version Up** from the menu bar. This changes the version number in the file that's rendered, for example, from Precomp1_v01_####.exr to Precomp1_v02_####.exr.
4. Render the Write node. Note that if the proxy mode is on and you have entered a proxy file name for the output node, Nuke will render a proxy image.
5. Select **File** > **Save New Version** (or **File** > **Save As**) to save the script with a new version number, for example, Precomp1_v02.nk instead of Precomp1_v01.nk.

To reload a revised precomp script:

1. Load the main comp.
2. Make sure the filename in the Precomp node's **file** field matches the current name of the precomp script.

   If the precomp script name has been versioned up, you can simply select the Precomp node and choose **Edit** > **Node** > **Filename** > **Version Up** from the menu bar. This changes the name of the script that is read in, for example, from Precomp1_v01.nk to Precomp1_v02.nk.

   If the precomp script has been saved with a different name or you want to use a different script as the precomp script, edit the filename in the **file** field.
3. In the Precomp properties, click **Reload**.

Collaborative Workflow Example

In this example, a compositor is responsible for the main comp that will be delivered to the client. A lightingTD is responsible for providing a multi-layered CG render for use in the main comp. The following describes how the compositor and lightingTD could use the Precomp node to enable a collaborative workflow.

To enable a collaborative workflow:

1. The compositor starts building the main comp.
2. The lightingTD does a render of the CG element and a comp of the layers, saving the script as cg_v01.nk.

Important *Note that we recommend rendering precomp images as exrs. This way, Nuke will write the hash value of the incoming node tree into the rendered file. If the precomp script changes so that the hashes won't match, Nuke will notify the user who can then update the resulting image. If you use a file format other than exr, the user will not get this notification and the rendered file is likely to become out of date.*

For more information on hash values, see "What is the hash value?" on page 531.

3. The compositor creates a Precomp node reading from the file cg_v01.nk, and continues working on the main comp.
4. The lightingTD continues to revise the CG render and the comp, versioning up the Write node and the precomp script .nk name. (See "To view and edit a precomp script:" on page 151.)

   When better results are achieved, the lightingTD notifies the compositor of the new and improved version of the precomp script.
5. The compositor versions up the Precomp node to the new, improved version and reloads the precomp script. (See "To reload a revised precomp script:" on page 151.)

Working with File Metadata

The nodes in the Metadata menu of the Toolbar let you work with information embedded in your images. This section gives instructions on their usage, teaching you to view, compare, edit, and render metadata.

Metadata in Nuke

Metadata is a set of information about an image embedded in the image file. This information may include the image's original bit depth, width, and height, for example. It can be attached to the file by the camera used to shoot the images, and/or edited later.

When Nuke loads an image, it reads in the metadata embedded in the image. The metadata is then passed down the node tree so you can view and use it at any point in your script. For example, you can reference metadata via expressions. You can also edit or delete existing metadata, add new metadata to a file, and write the resulting metadata out to files.

Note *Metadata for QuickTime files does not show gamma or bit depth.*

Note *When using a Merge node, you can choose which input's metadata to pass down the tree. In the Merge controls, set **metadata from** to either **A** or **B**.*

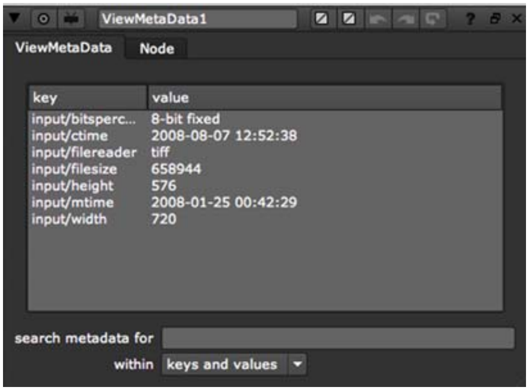
In the MetaData menu of the Toolbar, Nuke features five nodes that help you work with file metadata:

- ViewMetaData lets you inspect the metadata passed down by the input node. See "Viewing Metadata" on page 153.
- CompareMetaData lets you compare the metadata between two inputs and view the differences. See "Comparing Metadata Between Inputs" on page 154.
- ModifyMetaData lets you edit existing metadata in the input stream, add metadata into the stream, and delete metadata from the stream. See "Modifying Metadata" on page 154.
- CopyMetaData lets you copy metadata from one input to another and filter metadata to exclude some of it. See "Copying Metadata from One Input to Another and Filtering Metadata" on page 157.
- AddTimeCode lets you add a timecode to the metadata passed down by the input node. See "Adding a Timecode to Metadata" on page 157.

Viewing Metadata

To view metadata:

1. Select **MetaData** > **ViewMetaData** to insert a ViewMetaData node after the node whose metadata you want to inspect.
2. In the ViewMetaData properties, you can see a list of the metadata embedded in the image. This is divided into keys and their values.



key	value
input/bitsperc...	8-bit fixed
input/ctime	2008-08-07 12:52:38
input/filereader	tiff
input/filesize	658944
input/height	576
input/mtime	2008-01-25 00:42:29
input/width	720

search metadata for

within keys and values

3. To filter the lists of metadata, use the **search metadata for** field. For example, if you enter f in the **search metadata for** field, only the keys and values that include the letter f are displayed. By default, the search is done within both keys and values. If you want to limit to search to either keys or values only, set **within** to **keys only** or **values only**. For example, you can view metadata specific to the input's file format by entering the file format (for instance, dpx/) in the **search metadata for** field and setting **within** to **keys only**.

Note *When observing the creation time (input/ctime) of an image, Windows generally differs from Linux and Mac OS X. This is due to the different way in which the operating systems deal with file creation times.*

Once you know which keys exist on the input, you can reference them in expressions. See "Accessing Metadata via TCL Expressions" on page 159.

Comparing Metadata Between Inputs

To compare metadata between two inputs:

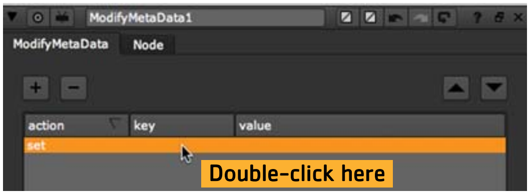
1. From the Toolbar, select **MetaData** > **CompareMetaData** to add a CompareMetaData node after the two nodes whose metadata you want to compare.
2. Connect the nodes you want to compare to the **A** and **B** inputs of the CompareMetaData node.

   A list of keys where there is a difference between the two inputs is shown in the CompareMetaData properties.

Modifying Metadata

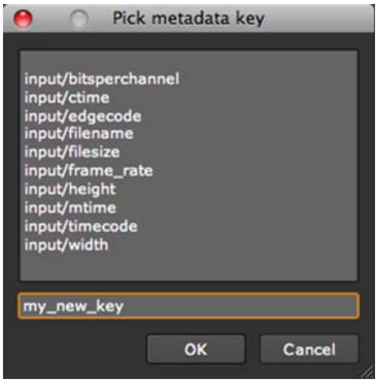
To add metadata:

1. Select **MetaData** > **ModifyMetaData** to insert a ModifyMetaData node after the node whose metadata you want to add a new key to.
2. In the ModifyMetaData controls, click on the plus (**+**) button. A placeholder appears in the metadata box.
3. Double-click on the placeholder under **key**.



The *Pick metadata key* dialog opens.

4. In the field at the bottom of the dialog, enter a name for the new key you want to add to the metadata. Click **OK**.


Pick metadata key

input/bitsperchannel
input/ctime
input/edgecode
input/filename
input/filesize
input/frame_rate
input/height
input/mtime
input/timecode
input/width

my_new_key

5. Double-click on the placeholder under **data** and enter a value for the new key.

   The new key and its value are added to the metadata being passed through.

To edit metadata:

1. Select **MetaData** > **ModifyMetaData** to insert a ModifyMetaData node after the node whose metadata you want to edit.
2. In the ModifyMetaData controls, click on the plus (**+**) button.

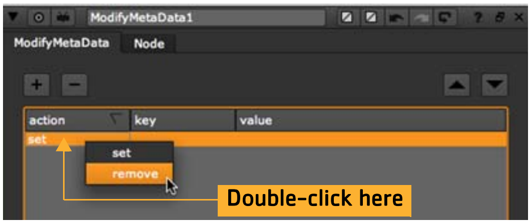
   A placeholder appears in the metadata box.
3. Double-click on the placeholder under **key**.

   The *Pick metadata key* dialog opens.
4. Pick the key whose name or value you want to edit and click **OK**.

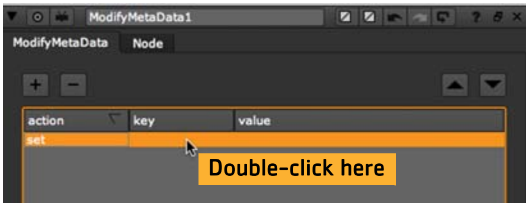
   The key is added to the ModifyMetaData properties.
5. In the ModifyMetaData properties, double-click on the key or its value and edit the information as required.

To remove metadata:

1. Select **MetaData** > **ModifyMetaData** to insert a ModifyMetaData node after the node whose metadata you want to edit.
2. In the ModifyMetaData properties, click on the plus (**+**) button. A placeholder is added to the metadata list.
3. Double-click on the placeholder under **action** and select **remove** from the menu that opens.

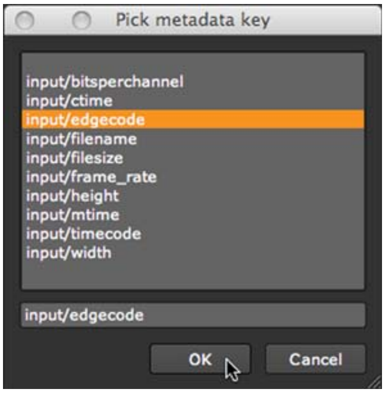

ModifyMetaData1
ModifyMetaData
Node
action
key
value
set
set
remove
Double-click here

4. Double-click on the placeholder under **key**.


ModifyMetaData1
ModifyMetaData
Node
action
key
value
set
Double-click here

The *Pick metadata key* dialog opens.

5. From the list of existing keys, select the key you want to remove and click **OK**.


Pick metadata key
input/bitsperchannel
input/ctime
input/edgecode
input/filename
input/filesize
input/frame_rate
input/height
input/mtime
input/timecode
input/width
input/edgecode
OK
Cancel

The node now removes the selected key from the metadata, as you can see if you view the metadata in the output.

To edit the list of actions in the ModifyMetaData properties:

- To perform a new action, click on the plus (**+**) button.
- To cancel an existing action, select it from the list and click on the minus (**–**) button. Note that this only affects the ModifyMetaData actions, and does NOT delete keys from the metadata embedded in the input image.
- To move an item up in the list, select it and click on the up arrow button.
- To move an item down in the list, select it and click on the down arrow button.

Copying Metadata from One Input to Another and Filtering Metadata

To copy metadata from one input to another and/or filter metadata:

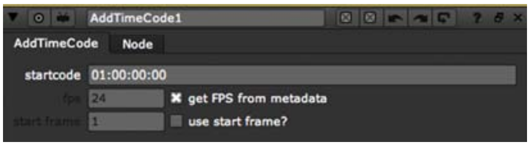
1. Select **MetaData** > **CopyMetaData** to insert a CopyMetaData node into your script.
2. Connect
   - the **Image** input to the node whose image you want to pass down the tree.
   - the **Meta** input to the node whose metadata you want to copy to the output.
3. Set **metadata from** to one of the following:
   - **Image+Meta** to add the metadata from the **Meta** input to the metadata from the **Image** input. If the inputs share any common metadata keys, the values taken from the Meta input override those taken from the Image input.
   - **Meta only** to only use the metadata from the Meta input.
   - **Meta+Image** to add the metadata from the **Image** input to the metadata from the **Meta** input. If the inputs share any common metadata keys, the values taken from the Image input override those taken from the Meta input.
   - **Image only** to only use the metadata from the Image input. This produces the same result as not using a CopyMetaData at all: both the image and metadata are taken from the Image input. However, this option can be useful if you want to filter the metadata to exclude some of it (see the next step for how to do this).
4. To filter the metadata taken from the inputs, use the **copy only** fields under **Meta filtering** and/or **Image filtering**. For example, if you enter **f** in the **copy only** field under **Meta filtering**, only the keys and values that include the letter f are copied from the Meta input. By default, the search is done within both keys and values. If you want to limit to search to either keys or values only, set **within** to **keys only** or **values only**. For example, you can copy metadata specific to the input's file format by entering the file format (for instance, **dpx/**) in the **copy only** field and setting **within** to **keys only**.

Adding a Timecode to Metadata

To add a timecode to metadata:

1. Select **MetaData** > **AddTimeCode** to insert an AddTimeCode node into your node tree.

   A timecode is added to the metadata being passed through. By default, the timecode is 01:00:00:00 on the first frame. It is updated throughout the frame range according to the input clip's playback speed, which in turn is controlled by the **fps** (frames per second) parameter in the Project Settings. If you change the **fps** value in the Project Settings, the timecode in the metadata is updated to reflect the change.



startcode 01:00:00:00
fps 24 ☒ get FPS from metadata
start frame 1 ☐ use start frame?

2. If you don't want the timecode on the start frame to be 01:00:00:00, enter a new timecode in the **startcode** field.
3. If you want to specify the playback speed manually rather than get it from the metadata and project settings, uncheck **get FPS from metadata** and enter a new value in the **fps** field.
4. If you want to specify a different start frame than the first frame, check **use start frame?** and enter a new value in the **start frame** field.

If you want to display the timecode on the image, insert a Text node after the AddTimeCode node and enter [timecode] in the **message** field. For more information on referencing metadata via expressions, see "Accessing Metadata via TCL Expressions" below.

Rendering Metadata

When rendering with the Write node, Nuke lets you write out metadata into the following file formats: EXR, CIN, DPX, and JPEG. You cannot write out metadata into any other formats.

When rendering metadata into an EXR file, you can use the **metadata** menu in the Write node controls to specify what to write out:

- **no metadata** - Do not write out any metadata, except for metadata that fills required header fields (for example, filename and bbox).
- **default metadata** - Write out the timecode, edgecode, frame rate, and exposure.
- **default metadata and exr/*** - Write out the timecode, edgecode, frame rate, exposure, and anything in **exr/**.
- **all metadata except input/*** - Write out all metadata, except anything in **input/**.
- **all metadata** - Write out all metadata.

When rendering any other file format than EXR, Nuke writes whatever metadata the file format header is known to support. Therefore, what is written out varies according to the file format.

Accessing Metadata via TCL Expressions

You can access metadata via TCL expressions in the following ways:

- To get a list of all keys in the incoming metadata, use the expression **[metadata]**. For example, if you add a Text node after an image and enter **[metadata]** in the **message** field, a list of all the keys in the incoming metadata appears on the image. The values of the keys are not displayed.
- To get the value of a particular key in the incoming metadata, use the expression **[metadata *key*]**. Replace ***key*** with the name of the key whose value you want to use. For example, to display the name and location of the image file on the image, add a Text node after the image and enter **[metadata input/filename]** in the **message** field.
- To get a filtered list of keys in the incoming metadata, use the expression **[metadata keys *filter*]**. Replace ***filter*** with whatever you want to use to filter the list. You can use asterisks (*****) as wildcards in your filter to substitute zero or more characters in the key names. For example, to get a list of all the keys with the letter **f** in them, use the expression **[metadata keys *f*]**. To get a list of all the keys starting with **input/**, use the expression **[metadata keys input/*]**.

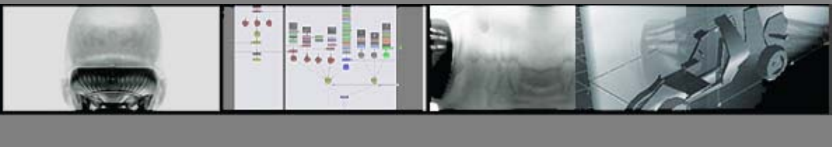
  By default, the keys are listed on separate lines. To change this, you can use **-s "*separator*"** to have the keys separated by a separator of your choice. Replace ***separator*** with whatever you want to appear between the different keys. For example, to get a list of all the keys starting with **input/** and separated by spaces, you can use **[metadata -s " " keys input/*]**. To get the same list separated by commas, use **[metadata -s ", " keys input/*]**.

By default, if you attempt to access metadata that does not exist in the stream, Nuke returns an empty string. To make this error instead, use the **-e** flag before other parameters.

For more information on using expressions, see "Expressions" on page 533.

Accessing Metadata via Python

You can also access metadata using the Python programming language. For more information, see "Accessing Node Metadata" on page 579.



REFERENCE

Each chapter in this section explains in detail a key feature of Nuke. You can use the section to familiarize yourself with the features you are particularly interested in, or to get answers to specific problems that arise during compositing.

For information on the features in NukeX, see “NukeX” on page 658.

Organisation of the Section

These are the topics covered by this section:

- Chapter 1, "Reformatting Elements", describes how you can reformat images through scaling, cropping, and pixel aspect adjustments. This chapter also covers working with bounding boxes.
- Chapter 2, "Channels", shows you how to manage image data using Nuke’s unique 1023-channel workflow.
- Chapter 3, "Merging Images", teaches you how to layer background and foreground elements together, create contact sheets, and copy rectangles from one image to another.
- Chapter 4, "Color Correction and Color Space", explains a broad sampling of Nuke’s many color correction tools.
- Chapter 5, "Transforming Elements", covers the tools for changing the size, location, and orientation of an image, including how to translate, scale, rotate, and skew elements in 2D and 3D space. This chapter also describes adding motion blur.
- Chapter 6, "Tracking and Stabilizing", shows how to generate and edit 2D tracking data for purposes of removing unwanted motion or applying it to other elements.
- Chapter 7, "Keying with Primatte", teaches you to use the blue/greenscreen keyer Primatte in Nuke.
- Chapter 8, "Keying With Keylight", teaches you to use the keyer tool Keylight in Nuke.
- Chapter 10, "Using RotoPaint", shows how to use Nuke’s RotoPaint node.
- Chapter 11, "Temporal Operations", explains how to apply time-based effects like clip retiming and motion blur. This chapter also explains how to perform editorial tasks, such as trimming and slipping.
- Chapter 12, "Warping and Morphing Images", teaches you to use the GridWarp and SplineWarp nodes to warp and morph images.
- Chapter 13, "Creating Effects", describes how you can create effects, such as star filter effects, on your images.

- Chapter 14, "Analyzing Frame Sequences", explains how to use the CurveTool node to analyze and match image sequences.
- Chapter 15, "3D Compositing", teaches you how to create and manipulate 3D scenes composed of objects, materials, lights, and cameras.
- Chapter 16, "Working with Stereoscopic Projects", describes how to composite stereoscopic material in Nuke.
- Chapter 17, "Previews and Rendering", teaches you how to write out image sequences from scripts in order to preview results or create final elements.
- Chapter 18, "Expressions", explains how to apply expressions or scripting commands to Nuke parameters.
- Chapter 19, "Setting Interface Preferences", discusses the available preference settings that you can use to make behavior and display adjustments to the interface.
- Chapter 20, "The Script Editor and Python", teaches you to use Python in Nuke's Script Editor to automate long procedures, for example.
- Chapter 21, "Configuring Nuke", explains how to set up Nuke for multiple artists working on the same project.

1 REFORMATTING ELEMENTS

This chapter teaches you how to reformat images through scaling, cropping, and pixel aspect adjustments. You will also learn to adjust bounding boxes to minimize processing and rendering times.

Reformatting Images

This section discusses scaling operations with specific regard to reformatting elements to match specific resolutions and pixel aspect ratios. Nuke includes at least two nodes designed for reformatting elements: Reformat, and Crop.

Using the Reformat Node

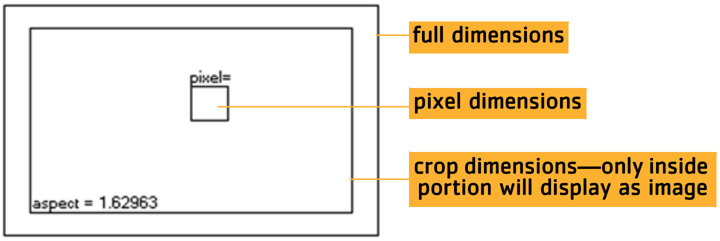
You can use the Reformat node for three different purposes:

1. To generate image sequences that match a desired image format in terms of both resolution and pixel aspect ratio (the width to height ratio of the format's individual pixels).
2. To create thumbnails (low resolution frames which you might post to the web in order to storyboard a sequence). The node scales the frame until it fits inside a rectangle whose dimensions you specify. It also sets pixel aspect ratio to one (square).
3. To scale images. The scale factor will be rounded slightly so that the output image has an integer number of pixels in the direction you choose in the Reformat node's controls.

Converting images to a desired image format

When you read in elements, Nuke stores their format settings and makes them available to the Reformat node. You can then apply one of the existing formats to your images, or create, edit, and delete formats yourself.

When creating a new format from scratch, you define the overall resolution, the cropped resolution (optional) and the pixel aspect ratio. As you define these parameters, the Reformat operator graphically displays them for you in the manner shown below.



To create a new output format:

1. Click **Transform** > **Reformat** to insert a Reformat node at an appropriate place in your script (generally before a Write node).
2. Connect a Viewer to the output of the Reformat node so you can see the effect of your changes.
3. Select **new** from the **output format** pulldown menu. The *New format* dialog appears.
4. Type a name for the new format in the **name** field.
5. In the **file size** fields, type the full output resolution (in pixels).
6. If you want to crop the full output resolution (for example, to create a letter box):
   - Check **image area**.
   - Increment the **x** field to define the left boundary of the crop. (The display updates to show you the left boundary of the crop relative to the full-size input.)
   - Increment the **y** field to define the bottom boundary of the crop.
   - Increment the **r** field to define the right boundary of the crop.
   - Increment the **t** field to define the top boundary of the crop.
7. If the destination display device for the image sequence uses nonsquare pixels, type the appropriate pixel aspect ratio in the **pixel aspect** field (for example, if your destination is a digital video display, type **1.1**).

Tip *You can also add formats to Nuke via entries to the* ***menu.py*** *file:*

1. Open the ***menu.py*** *file (located in same directory as your Nuke executable).*

2. Add an entry similar to the following example:

nuke.addFormat ("720 486 0 0 720 486 0.9 NTSC_video")

where the numbers specify, respectively, the format's full horizontal resolution, full vertical resolution, left crop position, bottom crop position, right crop position, top crop position, and pixel aspect ratio; and where the final text string designates the format's name.

3. Save and close the ***menu.py*** *file. The next time you launch Nuke the format will be available for selection from the Project settings dialog, Reformat node properties panel, and elsewhere.*

To edit a format:

1. Select the format you wish to edit from the **output format** pulldown list, then let the list close.
2. From the same list, select **edit**. The *Edit format* dialog appears.
3. Edit the **name**, **file size**, **image area**, and **pixel aspect** fields as necessary.
4. Click **OK** to save the changes to the format.

To delete a format:

1. Select the format you wish to delete from the **output format** pulldown list, then let the list close.
2. From the same list, select **delete**. The format is removed from the list.

To apply a format:

1. If necessary, click **Transform** > **Reformat** to insert a Reformat node at an appropriate place in your script (generally before a Write node).
2. Connect a Viewer to the output of the Reformat node so you can see the effect of your changes.
3. From the **type** pulldown menu, select **to format**.
4. Select the format you wish to apply from the **output format** pulldown list.
5. From the **resize type** field, choose the method by which you want to preserve or override the original aspect ratio. Select:
   - **width** to scale the original until its width matches the format's width. Height is then scaled in such a manner as to preserve the original aspect ratio.
   - **height** to scale the original until its height matches the format's height. Width is then scaled in such a manner as to preserve the original aspect ratio.
   - **fit** to scale the original until its smallest side matches the format's smallest side. The original's longer side is then scaled in such a manner as to preserve original aspect ratio.
   - **fill** to scale the original until its longest side matches the format's longest side. The input's shorter side is then scaled in such a manner as to preserve original aspect ratio.
   - **distort** to scale the original until all its sides match the lengths specified by the format. This option does not preserve the original aspect ratio, so distortions may occur.

6. When cropping the output, check **center** to position the crop area at the center of the frame.
7. Choose the appropriate filtering algorithm from the **filter** pulldown list (see "Choosing a Filtering Algorithm" on page 227).
8. When scaling an image with Key, Simon, and Rifmen filters, you may see a haloing effect which is caused by pixel sharpening these filters employ. If necessary, check **clamp** to correct this problem.

Creating thumbnails

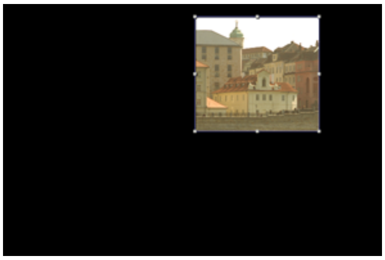
1. Click **Transform** > **Reformat** to insert a Reformat node at an appropriate place in your script (generally before a Write node).
2. Connect a Viewer to the output of the Reformat node so you can see the effect of your changes.
3. From the type pulldown menu, select **to box**.
4. In the **width** and **height** fields, type the output dimensions. The units are pixels.
5. Use the **resize type** pulldown menu to choose the method by which you preserve or override the original pixel aspect ratio. Select:
    - **width** to scale the original until its width matches the value in the **width** field. Height is then scaled in such a manner as to preserve the original aspect ratio (this means that the output you specified in **height** may not match the result).
    - **height** to scale the original until its height matches the value in the **height** field. Width is then scaled in such a manner as to preserve the original aspect ratio (this means that the output you specified in **width** may not match the result).
    - **fit** to scale the original until its smallest side matches the corresponding value in **width/height**. The longer side is then scaled in such a manner as to preserve the original aspect ratio.
    - **fill** to scale the original until its longest side matches the corresponding value in **width/height**. The smallest side is then scaled in such a manner as to preserve the original aspect ratio.
    - **distort** to scale the original until its sides match the values in the **width/height** fields. This option does not preserve the original aspect ratio, so distortions may occur.
6. Choose the appropriate filtering algorithm from the **filter** pulldown list (see "Choosing a Filtering Algorithm" on page 227).
7. When scaling an image with Key, Simon, and Rifmen filters, you may see a haloing effect which is caused by pixel sharpening these filters employ. If necessary, check **clamp** to correct this problem.

Scaling image sequences

1. Click **Transform** > **Reformat** to insert a Reformat node at an appropriate place in your script (generally before a Write node).
2. Connect a Viewer to the output of the Reformat node so you can see the effect of your changes.
3. From the type pulldown menu, select **scale**.
4. In the **scale** fields, enter scale factors for the width and the height. To scale both directions together using the same scale factor, click the **2** button.
5. Use the **resize type** pulldown menu to choose the method by which you preserve or override the original pixel aspect ratio. Select:
   - **width** to scale the original so that it fills the output width. Height is then scaled in such a manner as to preserve the original aspect ratio.
   - **height** to scale the original so that it fills the output height. Width is then scaled in such a manner as to preserve the original aspect ratio.
   - **fit** to scale the original so that its smallest side fills the output width or height. The longest side is then scaled in such a manner as to preserve the original aspect ratio.
   - **fill** to scale the original so that its longest side fills the output width or height. The smallest side is then scaled in such a manner as to preserve the original aspect ratio.
   - **distort** to scale the original so that both sides fill the output dimensions. This option does not preserve the original aspect ratio, so distortions may occur.
6. Choose the appropriate filtering algorithm from the **filter** pulldown list (see "Choosing a Filtering Algorithm" on page 227).
7. When scaling an image with Key, Simon, and Rifmen filters, you may see a haloing effect which is caused by pixel sharpening these filters employ. If necessary, check **clamp** to correct this problem.

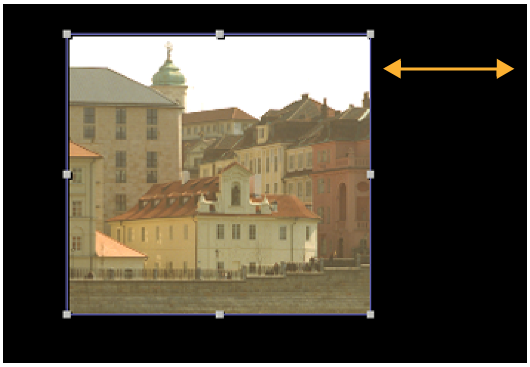
Cropping Elements

To *crop* a frame is to cut out the unwanted portions of the image area.

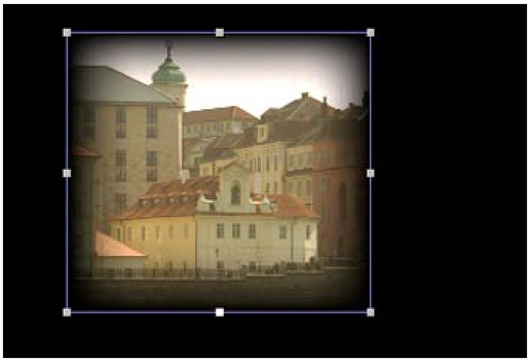


To crop elements:

1. Click **Transform** > **Crop** to insert a Crop node at an appropriate place in your script.
2. Connect a Viewer to the output of the Crop node so you can see the effect of your changes.
3. Define the crop boundaries:
   - In the Viewer, drag on any side of the frame to reposition it.



   - Or, in the Crop properties panel, increment or decrement the **box** field (**x** stands for left side, **y** for bottom side, **r** for right side, and **t** for top side).
4. To fill the cropped portion with black, check **black outside**. To fill the cropped portion by expanding the edges of the image, uncheck **black outside**. To adjust the image output format to match the cropped image, check **reformat**.
5. If you wish to vignette the edges of the cropped portion, increment the **softness** field.



Adjusting the Bounding Box

The bounding box defines the area of the frame that Nuke sees as having valid image data. The larger the bounding box is, the longer it takes Nuke to process and render the images. To minimize processing and rendering times, you can crop the bounding box. Occasionally, the bounding box may also be too small, in which case you need to expand it.

To adjust the bounding box, you can use the AdjBBox and CopyBBox nodes. The AdjBBox node lets you both crop and expand the bounding box edges, whereas with the CopyBBox node you can copy a bounding box from one input to another. If needed, you can also add a black outside edge to the bounding box using the BlackOutside node.

Resizing the Bounding Box

The AdjBBox node lets you expand or crop the edges of the bounding box by a specified number of pixels.

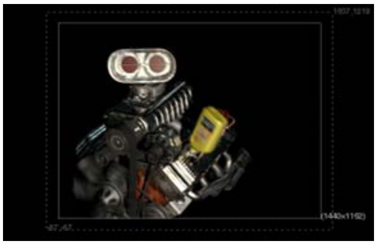


Figure 1.1: box with the AdjBBox node: An expanded bounding box.

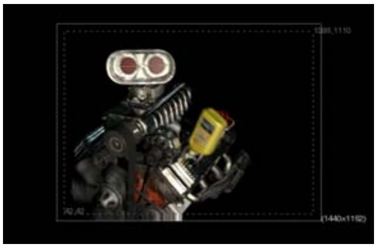


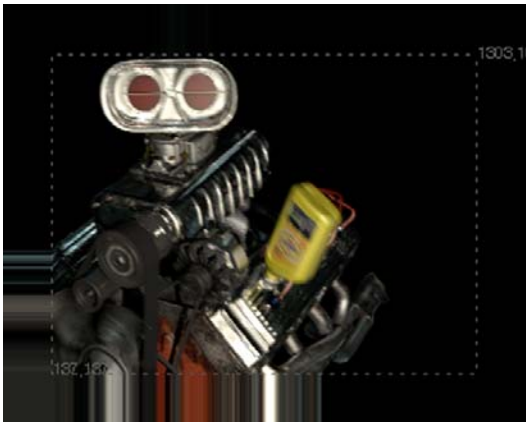
Figure 1.2: box with the AdjBBox node: A cropped bounding box.

For example, if you have an image with lots of black (0,0,0,0), you can adjust the bounding box to contain just the useful area so that Nuke won't waste time computing results where there is no change.

To resize the bounding box:

1. Select **Transform** > **AdjustBBox** to insert an AdjBBox node after the image whose bounding box you want to resize.
2. Connect a Viewer to the AdjBBox node, so you can see the effect of your changes.
3. In the AdjBBox controls, adjust the **Add Pixels** slider to increase or decrease the size of the bounding box. By default, 25 pixels are added to the edges of the bounding box.

Nuke expands or crops the edges of the bounding box. If the bounding box is cropped, whatever is outside the bounding box area gets replicated towards the edges of the image.



Copying a Bounding Box from One Input to Another

Some Nuke operations, such as a merge, can cause an expansion of the bounding box area because Nuke does not know that the extra area is going to be black or another constant color. Often, you can fix this by copying the bounding box from one of the inputs to the resulting image, thus cutting off this extra area. For this, you can use the CopyBBox node.

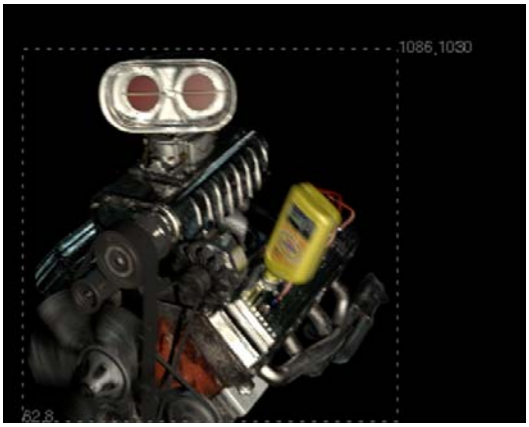


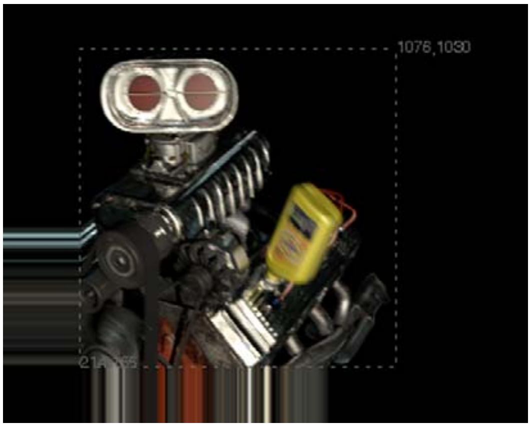
Figure 1.3: A bounding box from one input copied on top of another.

To copy a bounding box:

1. Select **Merge** > **CopyBBox** to insert a CopyBBox node after the node whose bounding box you want to use.

2. Connect the image whose bounding box you want to copy to the CopyBBox node's input A, and the image onto which you want to copy the bounding box to input B.

   Nuke copies the bounding box from input A to input B. Whatever is outside the copied bounding box area in image B gets replicated towards the edges of the image.



Adding a Black Outside Edge to the Bounding Box

If you adjust a bounding box with the AdjBBox or CopyBBox node, you may notice that whatever is outside the bounding box area gets replicated towards the edges of the image. If necessary, you can remove these replicated edge pixels and fill everything outside the bounding box area with black. To do this, use the BlackOutside node.

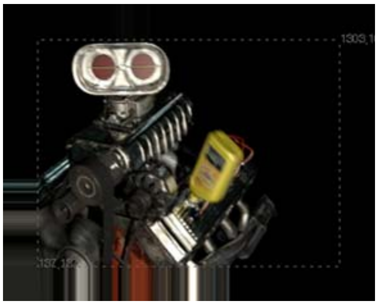


Figure 1.4: Using the BlackOutside node to add a black edge to the bounding box: A cropped bounding box with replicated edges.

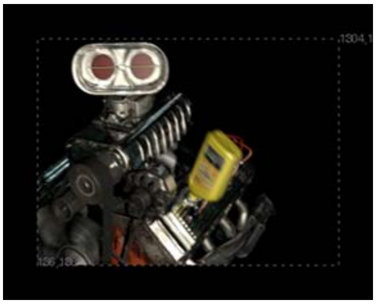


Figure 1.5: Using the BlackOutside node to add a black edge to the bounding box: The effect of the BlackOutside node.

To add a black outside edge to the bounding box:

1. Select the image whose edges outside the bounding box you want to fill with black.
2. Choose **Transform** > **BlackOutside** to add a BlackOutside node in an appropriate place in your script.

Nuke fills everything outside the bounding box area with black.

2 CHANNELS

Digital images generally consist of the four standard channels: red, green, blue, and alpha. Nuke allows you to create or import additional channels as masks, lighting passes, and other types of image data. A Nuke script can include up to 1023 uniquely named channels per compositing script.

For example, you can combine multiple render passes from a 3D scene—an image from the red, green, and blue channels, a depth mask (z-depth channel), a shadow pass, a specular pass, lighting passes, and multiple mattes all stored within one image sequence in your composite.

Overview

Before getting into the actual mechanics of channel management, it's important to review some basic concept about how Nuke processes channels.

Understanding Channels

Think of a channel as a container that contains image data. Once created or read into your composite, the image data stored in a channel is available downstream in the network until the value is replaced with something else or the channel is removed. The channel may even be "empty"—depending on where you reference it in the compositing network.

Understanding Channel Sets (Layers)

All channels in a script must exist as part of channel set (also called a *layer*). You're probably familiar with the default channel set—*rgba*—which includes the channels with pixel values of red, green, and blue, and also the alpha channel for transparency.

All channels in a composite must belong to at least one channel set. Some channels, like alpha, may be available in other sets, too. Channel names always include the channel set name as a prefix, like this: `set_name.channel_name.`

By default, every script has a channel set called **rgba**. When you first import an image element, Nuke automatically assigns its channels to the rgba set—that is, the image channels are named **rgba.red**, **rgba.blue**, **rgba.green**, and **rgba.alpha**.

The *rgba* set allows for the standard four-channel workflow of most node-based compositing systems. However, you're not limited to these four channels. You can create new channels and assign them to new channel sets up to the limit of 1023 channels per script.

Creating Channels and Channel Sets

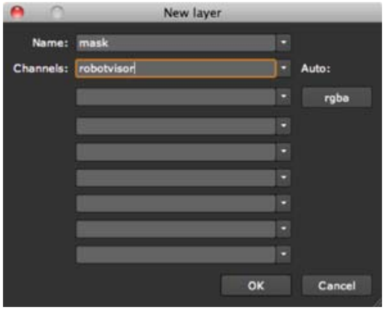
It's important to understand that many types of nodes allow you to direct their output to a specific channel and parent channel set. You have the option of processing these channels in each subsequent node, or leaving them unchanged.

Many nodes feature an **output** or **channels** setting, which lets you direct the output of the current node to a specific channel set and channel. You can also use the **output** or **channels** list to create new channel sets and channels.

Some nodes do not include an **output** or **channels** setting in their parameters. For these, you can connect other nodes, such as Channel Copy or Shuffle, to create and manage channel output in the node tree.

To create a new channel set and/or channel:

1. Open the properties panel for the node whose output will create the new channel.
2. From the **output** or **channels** pulldown list, select **new**.
3. Under **Name**, enter the name of the channel set, and under **Channels** the new channel name.



Name: mask
Channels: robotvisor

For example, as shown in the figure above, you would enter **mask** and **robotvisor** to create a new channel ("robotvisor") in the channel set named "mask." If the set does not already exist, Nuke will create it.

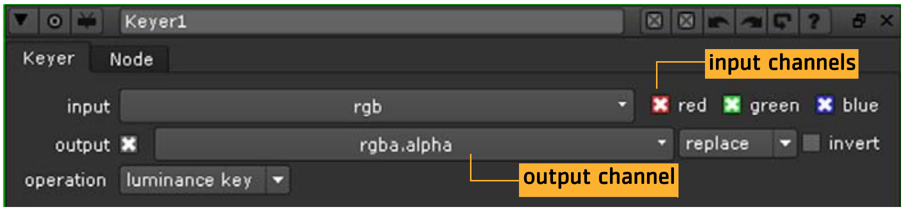
4. Click **OK**.

Note *You can also create new channels with the Shuffle and ShuffleCopy nodes. These are explained later, under Swapping Channels.*

Calling Channels

By default, most nodes in Nuke attempt to process the current channels in the rgba set and put output in those same channels. However, many nodes also contain an **input** list which lets you select the channels you want to process, and an **output** list to choose the channel(s) where the results should be stored.

Some nodes also contain **mask** controls and a mask input connector, which let you select a channel for use as a matte to limit operations such as color corrections. Using these mechanisms, you can point the output of almost any node in the script to any available channel.


Keyer1
Keyer
Node

input channels

input: rgb ☒ red ☒ green ☒ blue

output ☒ rgba.alpha replace ☐ invert

output channel

operation: luminance key

The script below attempts to clarify these concepts. Note the script generates six channels (though it could just as well generate 1023). The steps below describe how each channel was created.

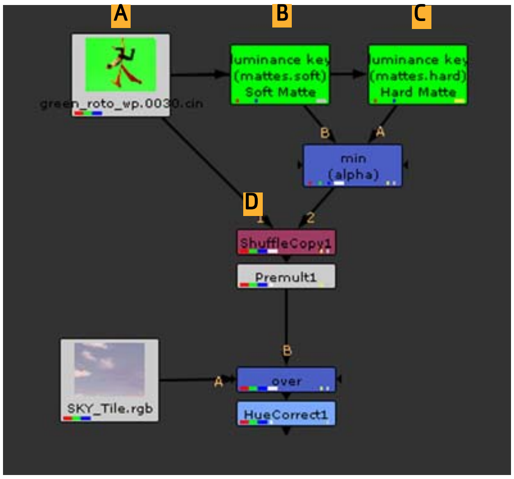

A
B
C
D
green_roto_wp.0030.cin
luminance key
(mattes.soft)
Soft Matte
luminance key
(mattes.hard)
Hard Matte
B
A
min
(alpha)
1
2
ShuffleCopy1
Premult1
B
A
SKY_Tile.rgb
over
HueCorrect1

Figure 2.1: A six-channel script.

A. The script reads in the foreground, creating three channels (**red**, **green**, and **blue**), which are by default assigned to the **rgba** set.

Channel count: 3

B. A low contrast key (**soft**) is pulled and assigned to a new channel set called **mattes**.

Channel count: 4

C. A high contrast key (**hard**) is pulled and also assigned to the mattes set.

Channel count: 5

D. The **mattes.hard** and **mattes.soft** channels are mixed to form the final matte (**alpha**), which is assigned to the **rgba** set.

Channel count: 6

Suppose now that you wanted to perform a color correction using the output of the Soft Matte node as a mask for the correction. There's no need to pipe the output from that Soft Matte node—it already exists in the data stream along with the other five channels that were created.

You simply attach a color correction node (for example, the HueCorrect node), then select the appropriate channel from the **mask** controls (in this example, **mattes.soft**). (Again, the **mattes** portion of the name indicates the parent channel set.)

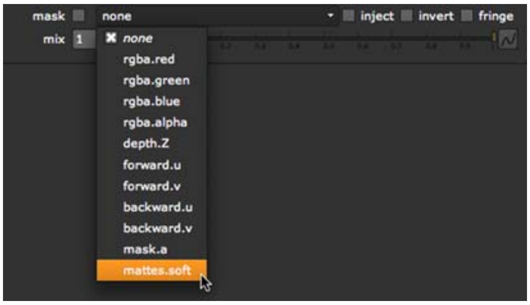

mask
none
inject
invert
fringe
mix
1
none
rgba.red
rgba.green
rgba.blue
rgba.alpha
depth.Z
forward.u
forward.v
backward.u
backward.v
mask.a
mattes.soft

Figure 2.2: Selecting a channel to mask color correction.

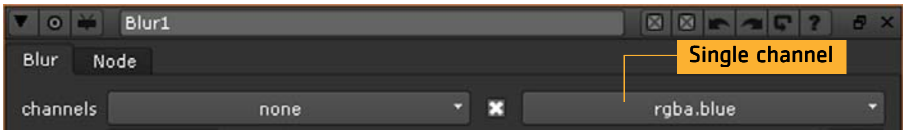
Selecting Input Channels

A node's **channels** field lets you select one or several channels for processing.

To select a single input channel:

1. Open the properties panel of the node into which you wish to feed a channel.
2. From the **channels** field, select **none**.

3. From the right most channel field—the one which typically calls the alpha channel—select the single channel you wish to process.

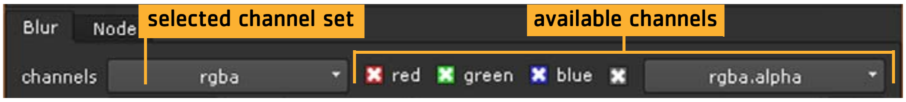


channels none ☒ rgba.blue

To select multiple input channels:

1. Open the properties panel of the node into which you wish to feed channels.
2. From the **channels** field, select the channel set containing the channels you wish to process.

   The set's channels appear with check boxes.



channels rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha

3. Uncheck those channels which you don't wish to process. The node will process all those you leave checked.

Selecting Masks

The **mask** controls in a node's properties panel let you select a single channel for use as a matte in a given process (typically, a color correction). The given process will thereafter be limited to the non-black areas of the selected channel.

You can use one of the script's existing channels as the matte, or attach a mask to the node with a mask input connector.

You can find mask input connectors on color correction and filter nodes, such as HueCorrect and Blur. At first, they appear as triangles on the right side of the nodes, but when you drag them, they turn into arrows labeled **mask** You connect them the same way as any other connectors. If you cannot see a mask input connector, open the node's properties panel and make sure **mask** is set to **none**.

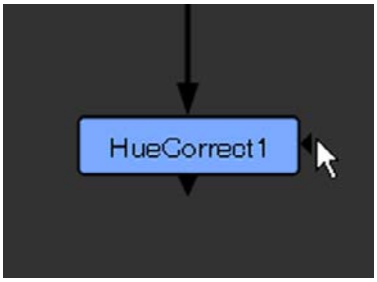


Figure 2.3: Using the mask input connectors that appear on some of the color, filter, channel, and merge nodes: Before dragging the connector.

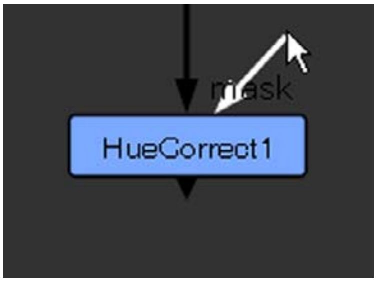
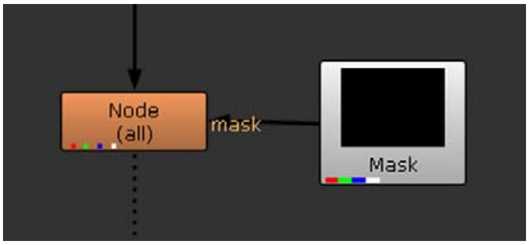


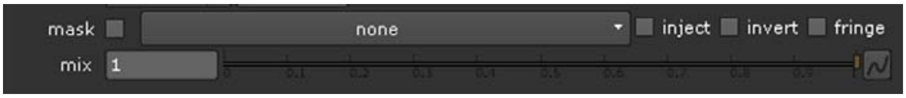
Figure 2.4: Using the mask input connectors that appear on some of the color, filter, channel, and merge nodes: When dragging the connector.

To select a channel for use as a matte from the mask input:

1. Connect a mask to the node with its mask input connector.

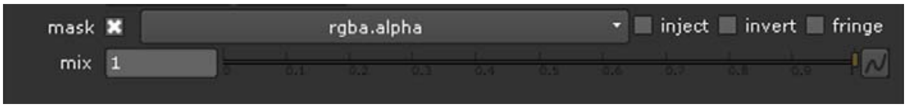


If you cannot see the mask input connector, open the node's controls and make sure **mask** is set to **none**.



mask ☐ none ☐ inject ☐ invert ☐ fringe
mix 1

By default, when a mask input is connected, the node uses the alpha channel from it as a matte.



mask ☒ rgba.alpha ☐ inject ☐ invert ☐ fringe
mix 1

2. If you don't want to use the alpha channel as the matte, select the channel you want to use from the **mask** pulldown menu.
3. If you want the mask from the **mask** input copied into the predefined **mask.a** channel, check **inject**. This way, you can use the last mask input again downstream. You can also set a stream of nodes to use **mask.a** as

the mask, and then change the masking of all of them by simply connecting a new mask into the mask input connector of the first node.

4. If necessary, check the **invert** box to reverse the mask.
5. If the overall effect of the node is too harsh, you can blend back in some of the input image by adjusting the **mix** slider.

To select a channel for use as a matte from the main input:

1. Make sure nothing is connected to the node's mask input connector. If you disconnect a mask input, the mask input connector disappears, as it is no longer being used.
2. Select the channel you want to use from the **mask** pulldown menu.
3. If necessary, check the **invert** box to reverse the mask.
4. If the overall effect of the node is too harsh, you can blend back in some of the input image by adjusting the **mix** slider.

Linking Channels

You can create expression links between channel and layer controls in various nodes. Since these controls aren't meant to be animated, you can't use the full range of Nuke expressions, nor can you use Python or TCL languages. You can link controls using the **Link menu** next to the control on the control panel:

1. Click the **Link menu** and select **Set link**. An Expression dialog opens.
2. Enter your expression in the Expression field and click **OK**. For more information on expressions, see "Expressions" on page 533.
3. You can edit an existing link by clicking the **Link menu** and selecting **Edit link**.
4. You can also **Ctrl/Cmd**+drag the **Link menu** to another control to create a link between the two.
5. To remove a link, click the **Link menu** and select **Remove link**.

Tracing Channels

You may have noticed that nodes visually indicate the channels which they are processing (that is, treating in some way) and passing (that is, conveying without any treatment). This is done via a system of colored rectangles, which allows you to trace the flow of channels throughout a script.

Look closer, for example, at the Over node. The wide rectangles indicate channels which Nuke processes (in this case, the red, green, blue, and alpha channels). The narrow rectangles indicate channels that Nuke passes onto the next node without processing (in this case, the **mattes.soft** and **mattes.hard**).

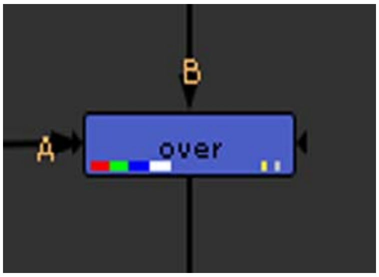


Figure 2.5: Visual confirmation of extra channels.

Renaming Channels

In the course of building your script, you may find it necessary to replace certain channels in a channel set.

To rename a channel:

1. Open the properties panel for a node which has the channel selected on the **channels**, **input**, or **output** pulldown list.
2. Click on the list where the channel is displayed and choose **rename**.

   The Rename Channel box appears.
3. Enter a new name for the channel and click **OK**.

Removing Channels and Channel Sets

When you are done using a channel set or a channel within a set, you may wish, for the sake of clarity, to remove it so that it is no longer passed to downstream nodes. Note that leaving channels in the stream will not itself cause them to be computed; only channels required are computed.

To remove a channel set or a channel within a set:

1. Click **Channel** > **Remove** to insert a Remove node at the appropriate point in your script.
2. In the Remove properties panel, select the channel set you wish to remove from the channels fields.
3. If you don't wish to remove the entire channel set, uncheck the boxes corresponding to the channels which you still wish to be able to call downstream.
4. Click **OK** to close the properties panel.

The channel set and/or the channels you removed will no longer be displayed in node parameters downstream from the Remove node.

Note *Removing channel sets and or channels does not free up space for the creation of new channels and sets. Once you create a channel it permanently consumes one of the script's 1023 available channel name slots. You are free, however, to rename channels and/or assign them new outputs.*

Swapping Channels

Nuke features two main nodes for channel swapping: Shuffle and Shuffle Copy. Shuffle lets you rearrange the channels from a single image (1 input) and then output the result to the next node in your compositing tree. Shuffle Copy lets you rearrange channels from two images (2 inputs) and output the result. Let's take a look at the ShuffleCopy node first.

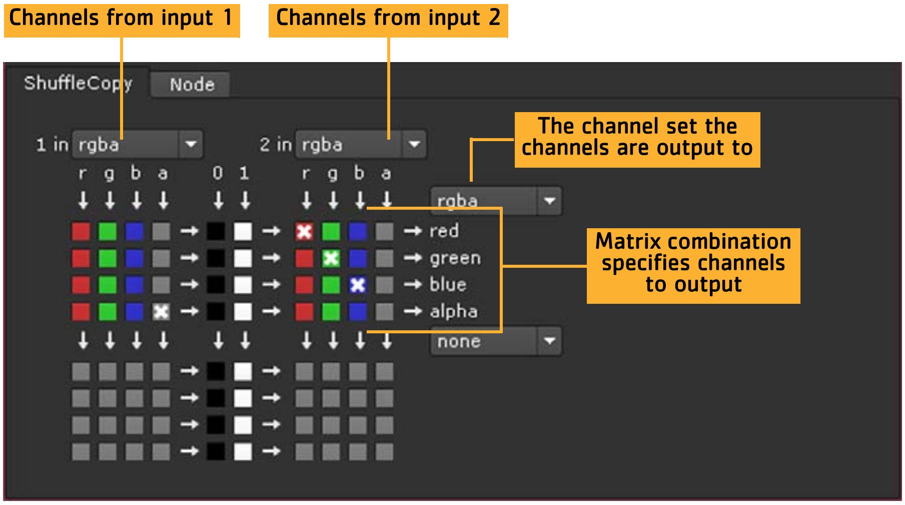

Channels from input 1
Channels from input 2
The channel set the channels are output to
Matrix combination specifies channels to output
ShuffleCopy
Node
1 in rgba
2 in rgba
r g b a
0 1
r g b a
rgba
red
green
blue
alpha
none

Figure 2.6: The Shuffle Copy matrix.

Channels from Input 1

The first group of channel boxes are the channels supplied by input 1 on the node. As shown above, the foreground element's default **rgba** set is selected.

Channels from Input 2

The second group of channel boxes are the channels supplied by input 2 on the node.

Channel Outputs

The combination of the boxes checked in the channel matrix create the list of channels that are output to the channel set selected in the top pulldown menu on the right.

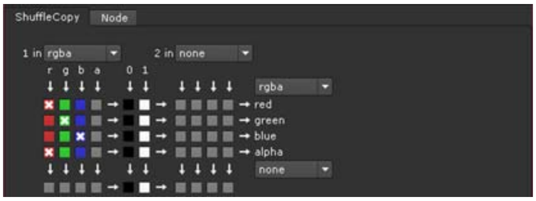
This four channel stream acts as *the second set of outputs* from the node. It allows you to output another four channels from the node, for a total of eight channels of output to match the possible eight channels of input.

In this case, this second set of outputs has not been utilized.

Tip *While not required, it's good practice to use the first set of outputs for swapping channels in the current data stream and the second set of outputs for creating new channels. This protects the default* ***rgba*** *set from unintentional overwriting, and makes it easier for other artists to understand the workings of your script.*

The basic process then for swapping channels is to first select your incoming channel sets from the **1 in** and **2 in** (or, in the case of the Shuffle node, the **in 1** and **in 2**) pulldown menus. Then, select your outgoing channel sets from the pulldown menus on the right. Then make the actual channel swaps by clicking on the resulting matrix.

For example, to take the simplest case, suppose, you wanted to copy red channel of the rgba set into its alpha channel. You would click on the matrix to create the following configuration.



1 in rgba
2 in none

Figure 2.7: Configuring the channel matrix.

You can see that the matrix makes use of the **r** channel (standing for **red**) twice. It goes out once as the red channel, and another time as the **alpha** channel.

Assigning Constants

The shuffle nodes also include parameters that let you assign white (1) or black (0) constants to any incoming channel. So, for example, to reset the alpha channel to a full-frame image, you would configure the matrix as follows:

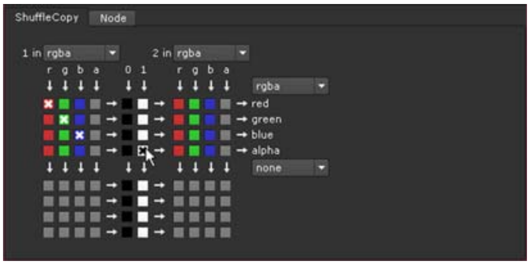


Figure 2.8: Assigning constants to channels.

Creating Swap Channel Sets

Finally, note that if the channel set to which you wish to output channels does not yet exist, you can create it using the **new** option on the pulldown menus on the right. Once you select the **new** option, you follow the same process for creating channel sets as is described in the "Creating Channels and Channel Sets" on page 173.

In Summary

The steps below summarize this discussion on swapping channels.

To swap channels:

1. Click **Channel** > **Shuffle** or **ShuffleCopy** to insert a Shuffle or Shuffle Copy node. Remember you use Shuffle when you only want to swap channels in a single upstream node, and Shuffle copy when you want to swap channels in two separate nodes, like a foreground and background branch.
2. Select the incoming channels from the **In 1** and **In 2** (optional) pull-down lists. You can select up to eight channels in this manner.
3. Select the channel sets to which you wish to direct the incoming channels from the pulldown lists on the right. You can select up to eight channels in this manner.
4. If the outgoing channel set to which you wish to direct channels does not yet exist, create it using the **new** option on the pulldown lists on the right.
5. Click as necessary on the resulting matrix to swap channels.

Tip *If you just need to copy a channel from one data stream into another, use **Channel** > **Copy**, instead of Shuffle Copy. Then, specify the channel to copy and the destination channel for output.*

3 MERGING IMAGES

With Nuke, you can merge images in a wide variety of ways. In this chapter, we teach you how to use the Merge, ContactSheet, and CopyRectangle nodes. The Merge node is used for layering multiple images together. The ContactSheet node lets you create a contact sheet that shows your frame sequence(s) lined up next to each other in a matrix. The CopyRectangle node can be used for copying a rectangular region from one image to another.

Layering Images Together with the Merge Node

The Merge node with its compositing algorithms allows you to control just how your images are combined.

Note *When using most of the available merge algorithms, Nuke expects premultiplied input images. However, with the **matte** operation you should use unpremultiplied images.*

To layer images with the Merge node:

1. Select **Merge** > **Merge** (or press M on the node graph) to insert a Merge node after the images you want to layer together.
2. Connect your images to the Merge node's **A** and **B** inputs.
3. If necessary, you can connect multiple A images to the Merge node. Once you have got the A and B inputs connected as instructed in step 2, drag more connectors from the left side of the Merge node to the images you want to use as additional A inputs.

   Each input is merged in the order connected, for example **A1**, **A2**, **A3**, **B**.
4. Connect a Viewer to the output of the Merge node so you can see the effect of your merge operation.
5. In the Merge node's controls, select how you want to layer the images together from the **operation** pulldown menu. The default and the most common operation is **over**, which layers input A over input B according to the alpha of input A. For descriptions of all the available operations, see "Merge Operations" below.
6. Using the **A channels** and **B channels** menus, select which channels to use from the A and B inputs and which channels to use as the A and B alpha. If you want to merge more channels than these and output them into the same channels, select them from the **also merge** pulldown menus and checkboxes.

7. From the **output** menu, select the channels you want to write the merge of the A and B channels to. Channels named in the **also merge** list are written to the same output channels.
8. If necessary, you can also adjust the following controls:
    - To select which input's metadata to pass down the tree, use the **metadata from** menu. For more information on file metadata, see "Working with File Metadata" on page 152.
    - To dissolve between the original input B image (at 0) and the full Merge effect (at 1), adjust the **mix** slider. A small light gray square appears on the node in the node graph to indicate that the full effect is not used.
    - If you want to mask the effect of the Merge operation, select the mask channel from the **mask** pulldown menus. To invert the mask, check **invert**. To blur the edges of the mask, check **fringe**.

      Note that you should not use the alpha of the inputs for the mask. It produces erroneous results (though the error is often hard to see); you can achieve better results by turning on alpha masking.
    - From the **Set BBox to** pulldown menu, select how you want to output the bounding box. The default is **union**, which combines the two bounding boxes. You can also choose **intersection** to set the bounding box to the area where the two bounding boxes overlap, **A** to use the bounding box from input A, or **B** to use the bounding box from input B.
    - By default, Nuke assumes that images are in linear color space. However, if you want to convert colors to the default 8-bit color space defined in the **LUT** tab of your project settings (usually, sRGB), check **Video colorspace**. The conversion is done before the images are composited together, and the results are converted back to linear afterwards. Any other channels than the red, green, and blue are merged without conversion.

      Checking this option can be useful if you want to duplicate the results you obtained from an application that uses the standard compositing math but applies it to non-linear images (for example, Adobe® Photoshop®). In this case, you typically also need to make sure **premultiplied** is not checked in your Read node controls.
    - By default, the same math is applied to the alpha channel as the other channels. However, according to the PDF/SVG specification, many of the merge operations (for example, overlay and hard-light) should set the alpha to (a+b - ab). This way, the input images remain unchanged in the areas where the other image has zero alpha. If you want to enable this, check **alpha masking**.

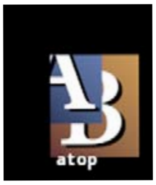
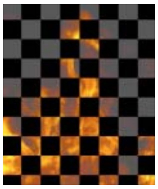
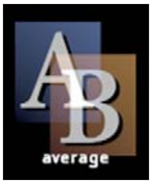
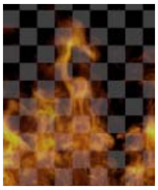
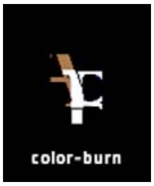
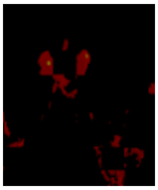
Merge Operations

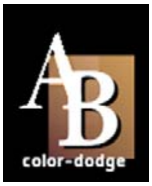
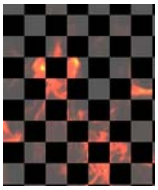
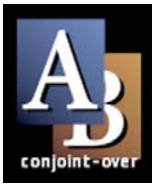
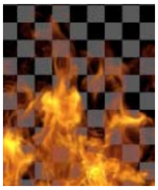
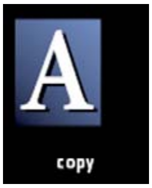
When layering images with the Merge node, you need to select a compositing algorithm that determines how the pixel values from one input are calculated with the pixel values from the other to create the new pixel values that are output as the merged image.

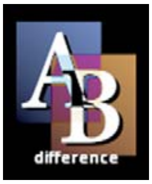
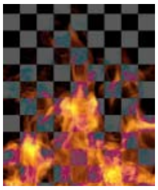
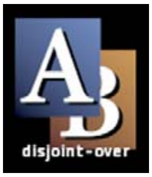
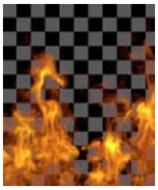
The **operation** menu in the Merge node's control panel houses a large number of different compositing algorithms, giving you great flexibility when building your composite. The available algorithms are listed in alphabetical order.

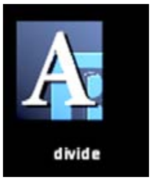
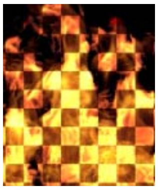
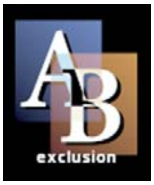
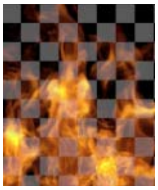
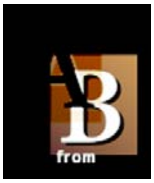
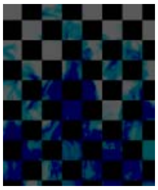
Tip *With many compositing algorithms available, it may sometimes be difficult to find what you're looking for in the **operation** menu. Luckily, there's a quick way of finding a particular operation. With the menu open, you can type a letter to jump to the first operator that starts with that letter. To move to the second operation that starts with the same letter, press the letter again. For example, to select the **soft-light** operation, open the menu and press **S** twice.*

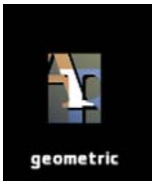
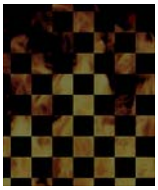
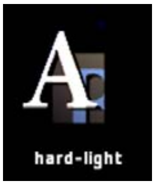
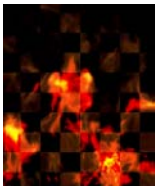
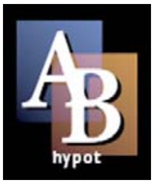
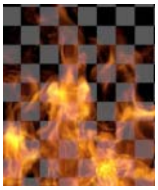
The following table describes each operation and its associated compositing algorithm. There are example images to illustrate the effects, one that combines the letters A and B to a merged image and another that has an image of fire merged with the familiar checkerboard. You may want to spend some time familiarizing yourself with each algorithm in order to be able to determine which operation to use in each situation.

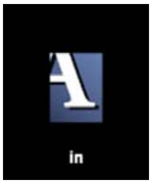
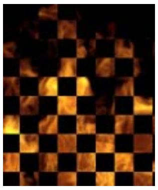
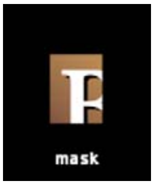
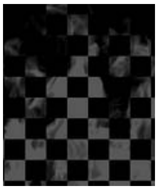
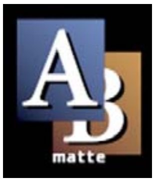
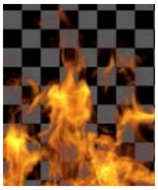
Operation	Algorithm	Description	Illustration	Example Uses
atop	Ab+B(1-a)	Shows the shape of image B, with A covering B where the images overlap.	 	
average	(A+B)/2	The average of the two images. The result is darker than the original images.	 	
color-burn	darken B towards A	Image B gets darker based on the luminance of A.	 	

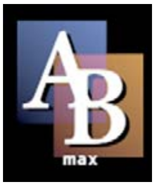
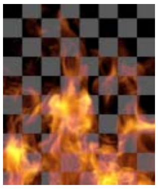
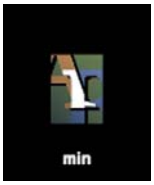
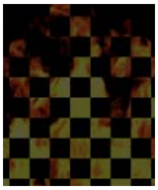
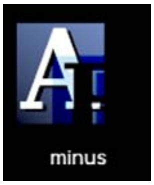
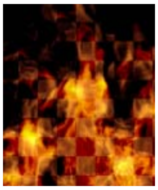
Operation	Algorithm	Description	Illustration	Example Uses
color-dodge	brighten B towards A	Image B gets brighter based on the luminance of A.	 	
conjoint-over	A+B(1-a/b), A if a>b	Similar to the over operation, except that if a pixel is partially covered by both a and b, conjoint-over assumes a completely hides b. For instance, two polygons where a and b share some edges but a completely overlaps b. Normal over will produce a slightly transparent seam here.	 	
copy	A	Only shows image A.	 	This is useful if you also set the **mix** or **mask** controls so that some of B can still be seen.

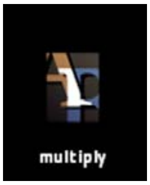
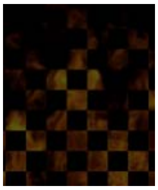
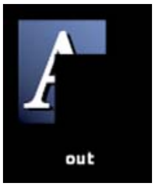
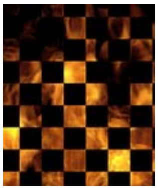
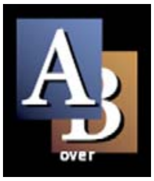
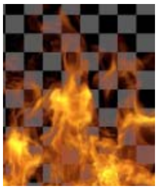
Operation	Algorithm	Description	Illustration	Example Uses
difference	abs(A-B)	How much the pixels differ. Also available from **Merge** > **Merges** > **Absminus**.	 	Useful for comparing two very similar images.
disjoint-over	A+B(1-a)/b, A+B if a+b<1	Similar to the over operation, except that if a pixel is partially covered by both a and b, disjoint-over assumes the two objects do not overlap. For instance, two polygons that touch and share an edge. Normal over will produce a slightly transparent seam here.	 	This can be useful if you want to merge element a over element b, and element a has element b already held out. For example, you may have a CG character whose hair, skin, and clothing are rendered separately so that each object has the other objects held out of the render. In this case, using the over operation would produce dark lines around the comped objects. This is because over does a hold-out of the background image, meaning the background will be held out twice.

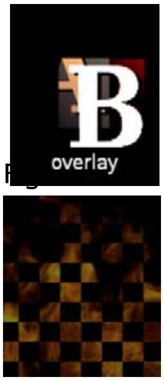
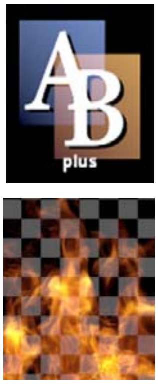
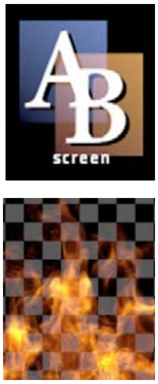
Operation	Algorithm	Description	Illustration	Example Uses
divide	A/B, 0 if A<0 and B<0	Divides the values but stops two negative values from becoming a positive number.	 	This does not match any photographic operation, but can be used to undo a multiply.
exclusion	A+B-2AB	A more photographic form of difference.	 	
from	B-A	Image A is subtracted form B.	 	

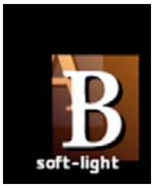
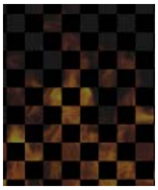
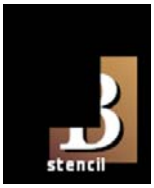
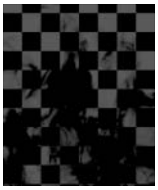
Operation	Algorithm	Description	Illustration	Example Uses
geometric	2AB/(A+B)	Another way of averaging two images.	 	
hard-light	multiply if A<0.5, screen if A>0.5	Image B is lit up by a very bright and sharp light in the shape of image A.	 	
hypot	sqrt(A*A+B*B)	Resembles the plus and screen operations. The result is not as bright as plus, but brighter than screen. Hypot works with values above 1.	 	This is useful for adding reflections, as an alternative to screen.

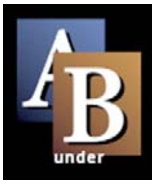
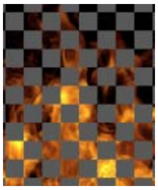
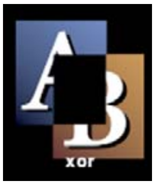
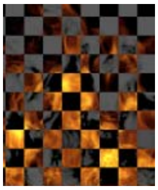
Operation	Algorithm	Description	Illustration	Example Uses
in	Ab	Only shows the areas of image A that overlap with the alpha of B. Also available from **Merge** > **Merges** > **In**.	 	Useful for combining mattes.
mask	Ba	This is the reverse of the in operation. Only shows the areas of image B that overlap with the alpha of A.	 	
matte	Aa+B(1-a)	Premultiplied over. Use unpremultiplied images with this operation. Also available from **Merge** > **Merges** > **Matte**.	 	

Operation	Algorithm	Description	Illustration	Example Uses
max	max (A,B)	Takes the maximum values of both images. Also available from **Merge** > **Merges** > **Max**.	 	This is a good way to combine mattes and useful for bringing aspects like bright hair detail through.
min	min (A,B)	Takes the minimum values of both images. Also available from **Merge** > **Merges** > **Min**.	 	
minus	A-B	Image B is subtracted from A.	 	

Operation	Algorithm	Description	Illustration	Example Uses
multiply	AB, A if A<0 and B<0	Multiplies the values but stops two negative values from becoming a positive number. Also available from **Merge** > **Merges** > **Multiply**.	 	Used to composite darker values from A with the image of B - dark gray smoke shot against a white background, for example. This is also useful for adding a grain plate to an image regrained with F_Regrain.
out	A(1-b)	Only shows the areas of image A that do not overlap with the alpha of B. Also available from **Merge** > **Merges** > **Out**.	 	Useful for combining mattes.
over	A+B(1-a)	This is the default operation. Layers image A over B according to the alpha of image A.	 	This is the most commonly used operation. Used when layering a foreground element over a background plate.

Operation	Algorithm	Description	Illustration	Example Uses
overlay	multiply if B<0.5, screen if B>0.5	Image A brightens image B.		
plus	A+B	The sum of image A and B. Also available from **Merge** > **Merges** > **Plus**. Note that the plus algorithm may result in pixel values higher than 1.0.		Useful for compositing laser beams, but you're better off not using this one for combining mattes.
screen	A or B ≤1? A+B-AB : max(A,B)	If A or B is less than or equal to 1 the screen else use the maximum example, resembles Plus. Also available from **Merge** > **Merges** > **Screen**.		This is useful for combining mattes and also for adding laser beams.

Operation	Algorithm	Description	Illustration	Example Uses
soft-light		Image B gets lit up. Not as extreme as the hard-light operation.	 	
stencil	B(1-a)	This is the reverse of the out operation. Only shows the areas of image B that do not overlap with the alpha of A.	 	

Operation	Algorithm	Description	Illustration	Example Uses
under	A(1-b)+B	This is the reverse of the over operation. Layers image B over A according to the matte of image B.	 	
xor	A(1-b)+B(1-a)	Shows both image A and B where the images do not overlap.	 	

Tip *If you have used older versions of Nuke, you may have seen Merge operations called diagonal, nl_over, and nl_under. Diagonal has been renamed and is now called hypot. To get the results of nl_over and nl_under, you can check* ***Video colorspace*** *and use over and under.*

Generating Contact Sheets

In order to demonstrate, document or manage what you are doing for a project, it can be useful to generate a contact sheet that shows your frame sequence(s) lined up next to each other in a matrix. For this, you can use the ContactSheet node. It generates a contact sheet from all its inputs or from the frames of one input.

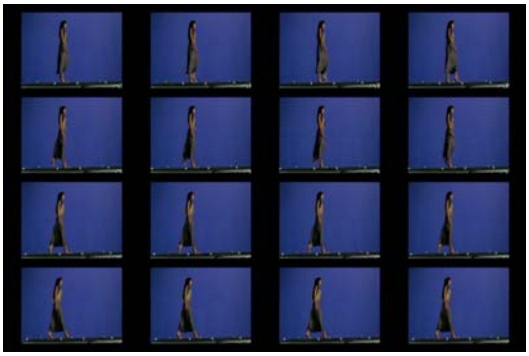
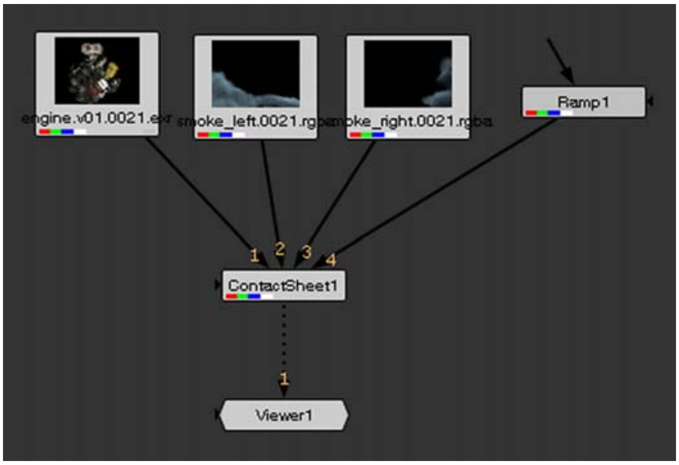
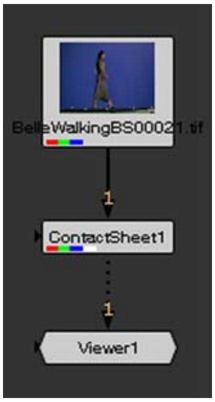


Figure 3.2: A contact sheet generated from the frames of one image sequence.

To generate a contact sheet:

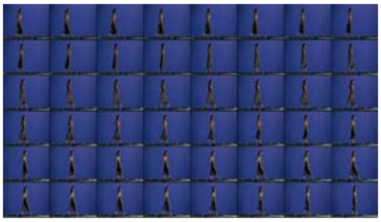
1. Select **Merge** > **ContactSheet** to insert a ContactSheet node in your script.
2. Connect the image(s) you want to include in your contact sheet to the numbered input(s) of the ContactSheet node. If you want to include several different image sequences in the contact sheet, use multiple inputs. If you want the contact sheet to include the frames of just one image sequence, use only one input.

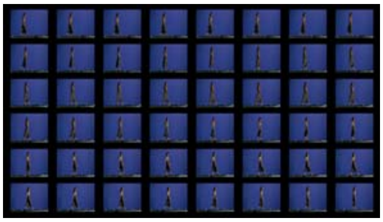

engine.v01.0021.exr
smoke_left.0021.rgba
smoke_right.0021.rgba
Ramp1
1
2
3
4
ContactSheet1
1
Viewer1


BelleWalkingBS00021.tif
1
ContactSheet1
1
Viewer1

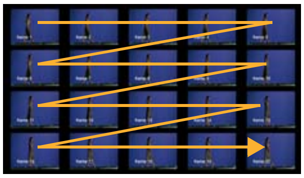
3. Connect a Viewer to the ContactSheet node so you can see the effect of your changes.
4. In the ContactSheet properties, define the **Resolution** (width and height) of the entire contact sheet in pixels.
5. If you want to create a contact sheet from the frames of one input, check **Use frames instead of inputs**. In the **Frame Range** field, define the frame range you want to include in the contact sheet.

6. In the **rows/columns** field, specify into how many rows and columns you want to arrange the input images or frames.
7. To adjust the size of the gaps between the images in the contact sheet, increment or decrement the **gap** value.

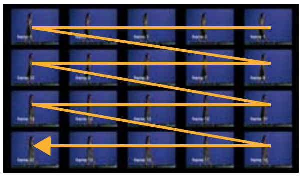

The gap value set to 0


The gap value set to 50

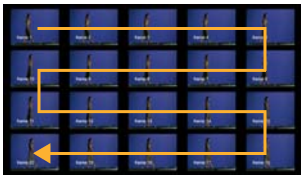
8. From the **Row Order** and **Column Order** menus, choose how you want to order the images or frames in the contact sheet:


Row Order: TopBottom

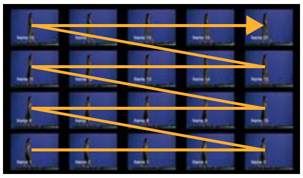
Column Order: LeftRight


Row Order: TopBottom

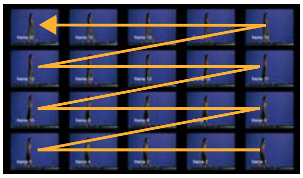
Column Order: RightLeft


Row Order: TopBottom

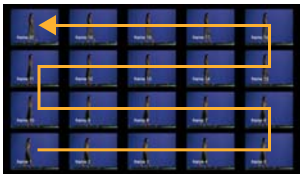
Column Order: Snake


Row Order: BottomTop

Column Order: LeftRight


Row Order: BottomTop

Column Order: RightLeft


Row Order: BottomTop

Column Order: Snake

Tip *If you want to add any text, such as the frame number, on top of the images in the contact sheet, insert a Text node between the input image(s) and the ContactSheet node.*

Copying a Rectangle from one Image to Another

With the CopyRectangle node, you can copy a rectangle from one input on top of another.

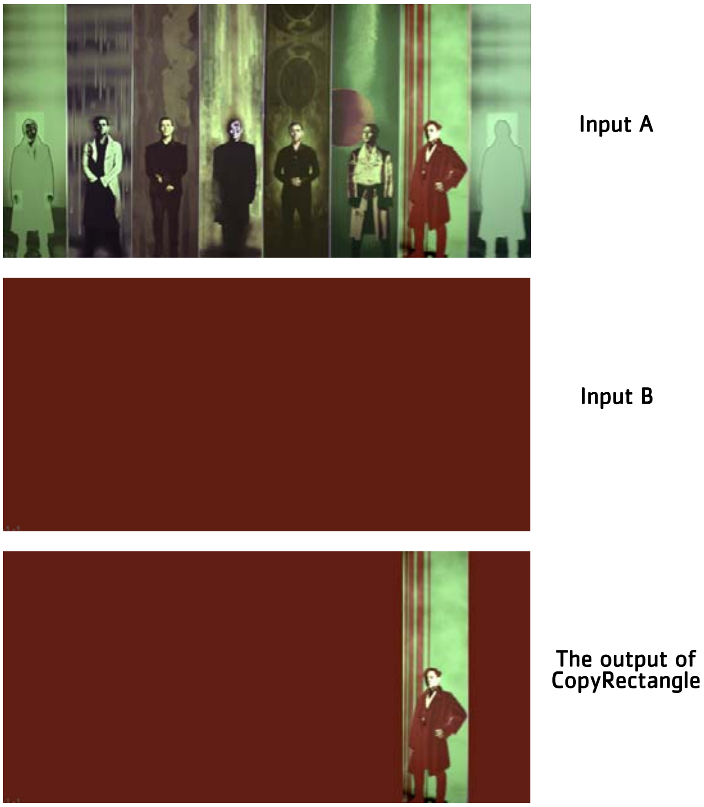

Input A
Input B
The output of CopyRectangle

Figure 3.3: Using CopyRectangle to copy a rectangular region from input A onto input B.

The CopyRectangle node can also be used to limit effects, such as color corrections, to a small region of an image. To do so, you need to use the same image in both input A and B and only perform the color correction on one input.

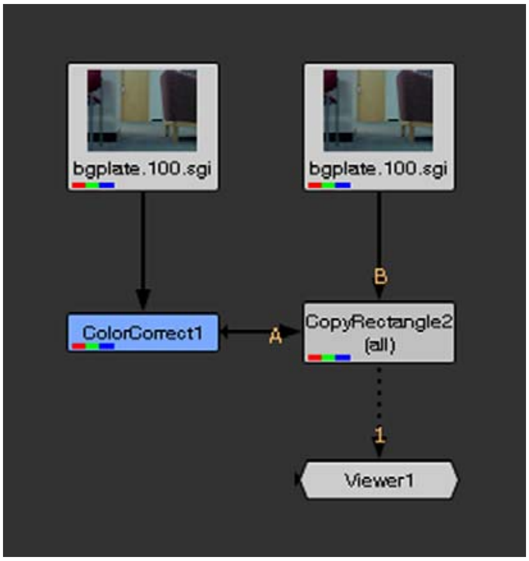

bgplate.100.sgi
bgplate.100.sgi
B
ColorCorrect1
A
CopyRectangle2
(all)
1
Viewer1

Figure 3.4: A rectangle from input A color corrected and copied on top of input B.

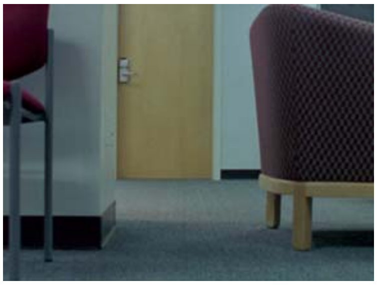


Figure 3.5: Using the CopyRectangle node to limit a color correction to a rectangular region: The original image.

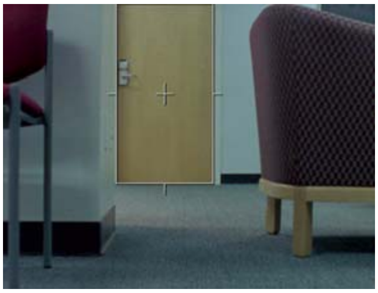


Figure 3.6: Using the CopyRectangle node to limit a color correction to a rectangular region: Defining a rectangle with CopyRectangle.

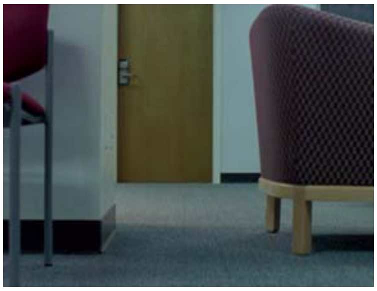
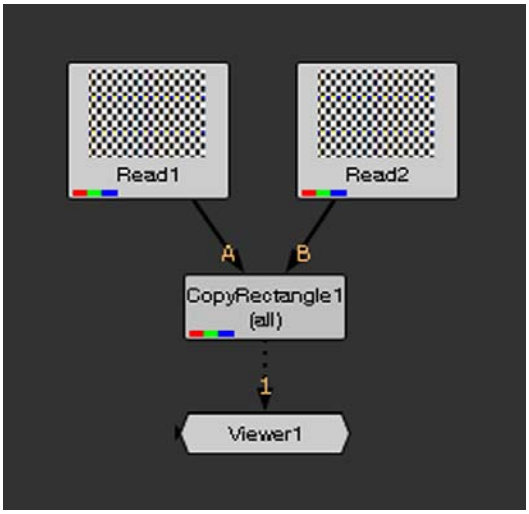


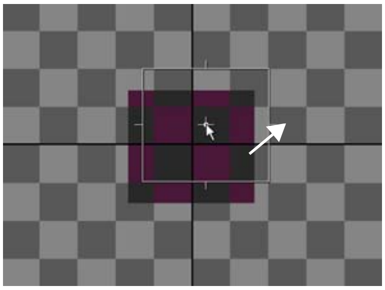
Figure 3.7: Using the CopyRectangle node to limit a color correction to a rectangular region: The color corrected rectangle on top of the original image.

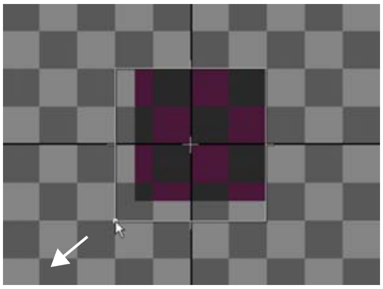
To copy a rectangle from one image to another:

1. Select **Merge** > **CopyRectangle** to insert a CopyRectangle node after the image that has a region you want to copy (input A) and the image you want to copy the region to (input B). Create the following setup:

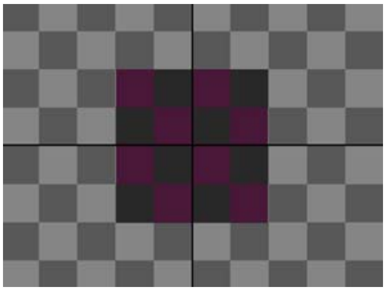


2. In the CopyRectangle controls, use the **channels** pulldown menu to select the channels you want to copy from input A.
3. To define the rectangle you want to copy, resize and reposition the CopyRectangle overlay in the Viewer. Drag the center of the overlay to reposition, and the edges to resize. If you cannot see the overlay in the Viewer, open the CopyRectangle properties panel and double-click on the Viewer node in the Node Graph.

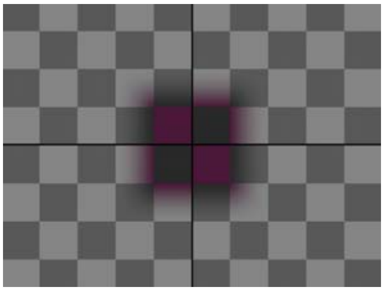

repositioning the rectangle


resizing the rectangle

4. To control how soft the edges of the rectangle seem, adjust the **softness** slider. The higher the value, the softer the edges.



low softness value



high softness value

5. To dissolve between the full CopyRectangle effect and input B, adjust the **mix** slider.

4 COLOR CORRECTION AND COLOR SPACE

This chapter explains how to use Nuke's color correction nodes to adjust the appearance of the images in your composites. Specifically, you'll learn how to:

- Make tonal adjustments.
- Make basic contrast, gain, gamma, and offset adjustments.
- Make hue, saturation, and value adjustments.
- Apply masks to color corrections.
- Convert elements into nonnative color spaces.
- Apply grain.

These topics provide a good overview of Nuke's color-correction nodes; however not all options are covered here. Look to Nuke's online help for instructions on using the other nodes found under the Color icon in the toolbar.

Making Tonal Adjustments

Defining tonal range (the blackpoint, whitepoint, and neutral value) is typically the first step in color correcting a clip. Tonal range adjustments often improve contrast, but more importantly, they set the stage for subsequent color corrections by properly dividing the colorspace into shadow, midtone, and highlight regions.

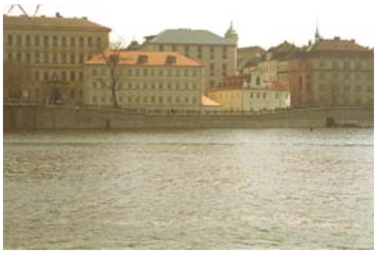


Figure 4.1: Before tonal adjustment.

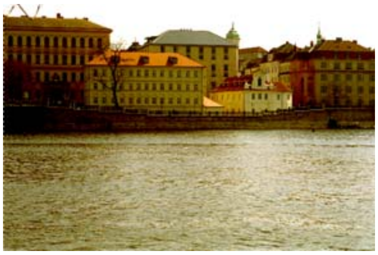


Figure 4.2: After tonal adjustment.

Several of Nuke's color correction effects offer tonal adjustment tools. Of these, Grade and Histogram are probably the most intuitive to operate.

Using Histograms

The properties panel for the Histogram node includes a window that graphs the number of pixels at each brightness level. This is a useful gauge to see whether an image has a good distribution of shadows, midtones, and highlights.

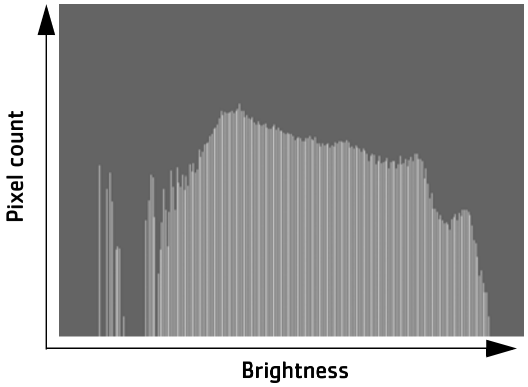


Figure 4.3: The histogram maps the distribution of shadows, midtones, and highlights.

To define tonal range with the Histogram node:

1. Click **Color** > **Histogram** to insert a Histogram node at the appropriate place in your script.
2. Connect a Viewer to the output of the Histogram node so you can see the effect of your changes.
3. Drag the leftmost **input range** slider till it roughly lines up with the initial boundary of the histogram.
4. Drag the rightmost **input range** slider till it roughly lines up with the final boundary of the histogram.
5. Drag the middle input range slider to define the midtone, or neutral, value.

Sampling White and Black Points

The Grade node lets you define white and black points by sampling pixels from a Viewer frame.

To define tonal range with Grade:

1. Click **Color** > **Grade** to insert a Grade node at the appropriate place in your script.
2. Connect a Viewer to the output of the Grade node so you can see the effect of your changes.
3. In the Grade properties panel, use the **channels** pulldown list to select the channels you wish to process.
4. Click the **blackpoint** parameter's color swatch.

   The eye dropper icon appears.

5. In the Viewer, press **Ctrl/Cmd+Shift** while clicking on the pixel you want to define as the blackpoint (typically the darkest pixel).
6. Click the **whitepoint** parameter's color swatch. The eye dropper icon appears.
7. In the Viewer, press **Ctrl/Cmd+Shift** while clicking on the pixel you want to define as the white point (typically the lightest pixel).

Making Basic Corrections

Adjustments to contrast, gamma, gain, and offset often comprise the bulk of the work in color correction. Some artists prefer to makes these adjustments via sliders; others prefer curves (which represent the range of color values in an image.) Nuke's ColorCorrect and ColorLookup nodes offer tools to suit either preference.

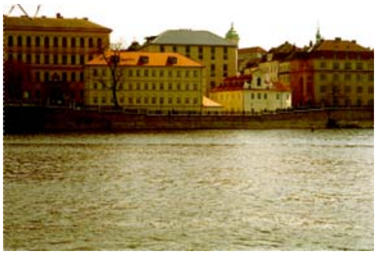


Figure 4.4: Original.

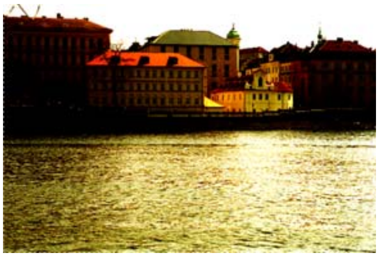


Figure 4.5: Contrast boost.

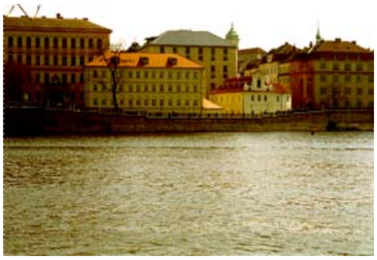


Figure 4.6: Original.

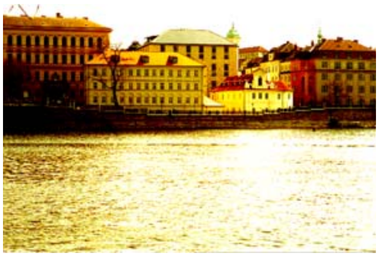


Figure 4.7: Gain boost.

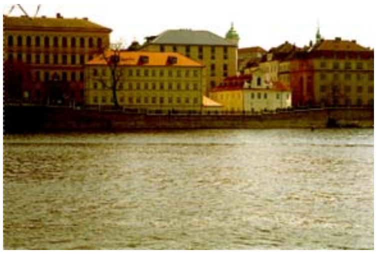


Figure 4.8: Original.

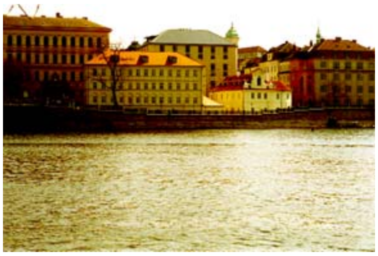


Figure 4.9: Gamma boost.

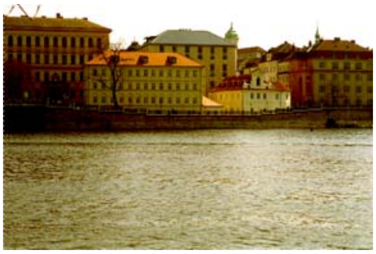


Figure 4.10: Original.

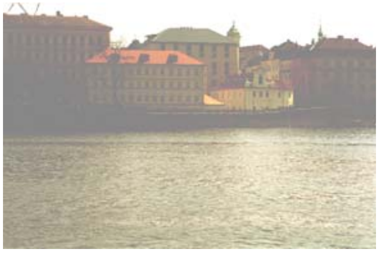


Figure 4.11: Offset boost.

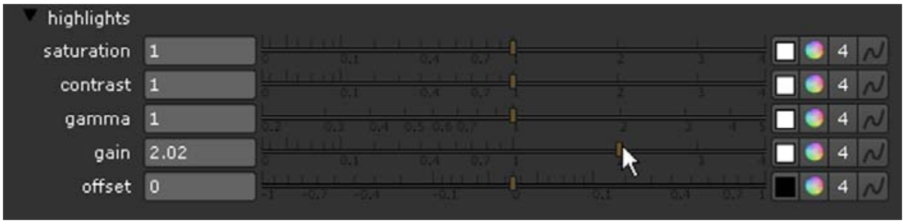
Using Sliders

The ColorCorrect node is particularly convenient for making quick adjustments to contrast, gamma, gain, and offset. A single window houses sliders for all these basic corrections and allows you to apply these to a clip's master (entire tonal range), shadows, midtones, or highlights.

To adjust contrast, gain, gamma or offset with the ColorCorrect node:

1. Click **Color** > **ColorCorrect** (or press **C**) to insert a ColorCorrect node at the appropriate place in your script.
2. Connect a Viewer to the output of the ColorCorrect node so you can see the effect of your changes.
3. In the ColorCorrect properties panel, use the **channels** pull-down list to select the channels you wish to process.
4. Drag the slider appropriate to the region you want to affect an operation you want to apply.

   For example, to brighten the images highlights, you would drag on the **highlights gain** slider.

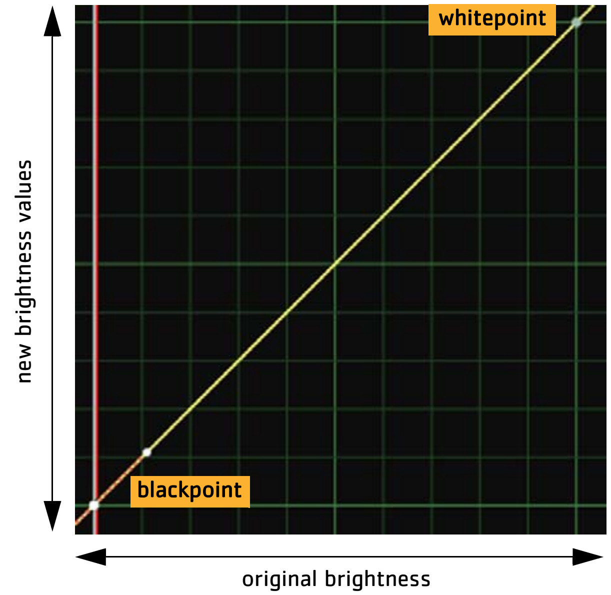


highlights
saturation 1
contrast 1
gamma 1
gain 2.02
offset 0

5. Remember too that you can use the color sliders to apply any of the corrections on a per channel basis.

Using Color Curves

If you prefer to work with color curves, you can use the ColorLookup node to make contrast, gamma, gain, and offset adjustments (and, in fact, many others). *Color curves* refer to line graphs of a given color channel's brightness. The horizontal axis represents the channel's original, or input, values, and the vertical axis represents the channel's new, or output values.



As Figure 4.12 shows, you can edit the ColorLookup node's color curves to make all of the types of corrections that are possible through the ColorCorrect node—and you can generally make these corrections with more flexibility and precision than is possible with sliders.

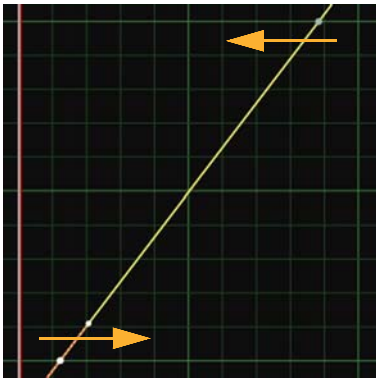


Figure 4.12: Corrections through the ColorLookup node's color curves: Contrast boost.

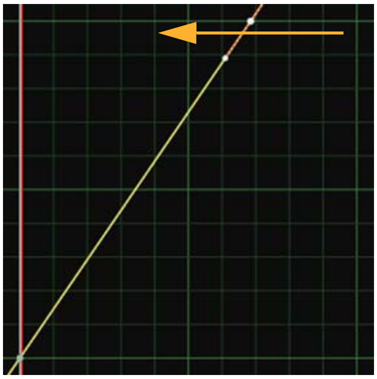


Figure 4.13: Corrections through the ColorLookup node's color curves: Gain boost.

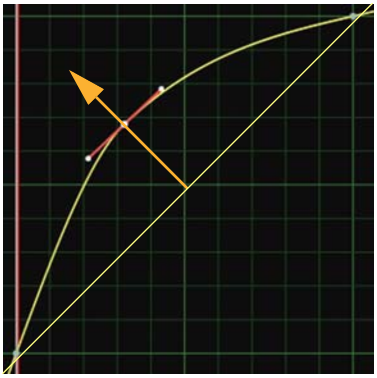


Figure 4.14: Corrections through the ColorLookup node's color curves: Gamma boost.

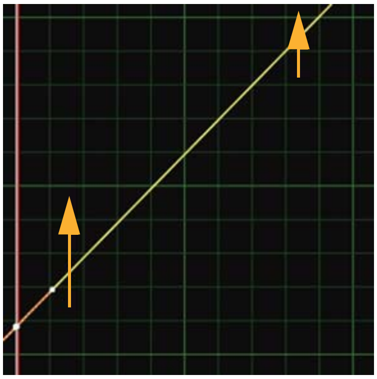


Figure 4.15: Corrections through the ColorLookup node's color curves: Offset boost.

To make basic corrections with the ColorLookup node:

1. Click **Color** > **ColorLookup** to insert a ColorLookup node at the appropriate place in your script.
2. Connect a Viewer to the output of the ColorLookup node so you can see the effect of your changes.
3. In the ColorLookup properties panel, click **red**, **green**, **blue**, or **alpha** if you want to limit the subsequent operations to a particular channel.

   You can select multiple curves in order to edit one curve with reference to another. Otherwise, select the **master** curve (which represents all channels).
4. To speed up the color calculation, the ColorLookup node uses a precomputed lookup table between 0 and the value specified in the **range** field. You can adjust the **range** value, or uncheck the **use precomputed table** box if necessary to tell ColorLookup to not use a precomputed table.
5. In the Viewer, drag the cursor over the pixels you want to sample for the correction. In the ColorLookup properties panel, press **Ctrl+Alt** (**Cmd+Alt** on a Mac) while clicking on the curve to set points at the places where the red, green, and blue lines intersect with the color curve.

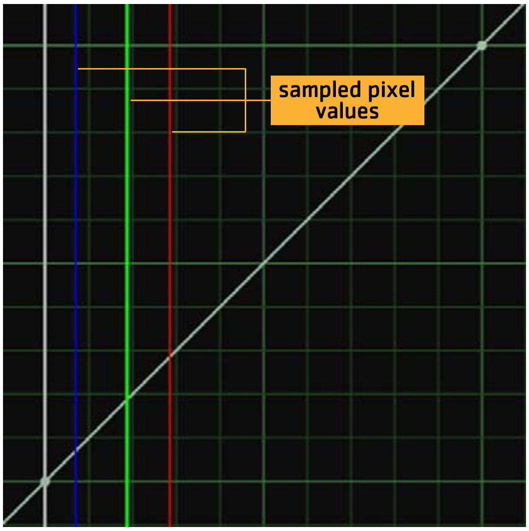


Figure 4.16: Viewing values from a sampled color.

6. Edit the position of the points and adjust the tangent handles to adjust the curve shape for the color correction.

   As an alternative to steps 4 and 5, you can use the **source** control to pick a source color for adding points. Then, use **target** to pick a destination color. Finally, do one of the following:

   - Click **Set RGB** to add points on the red, green, and blue curves, mapping **source** to **target**.
   - Click **Set RGBA** to add points on the red, green, blue, and alpha curves, mapping **source** to **target**.
   - Click **Set A** to add points on the alpha curve, mapping **source** to **target**.

   You can use these controls to match shadow, midtone, and highlights on two plates, for example. Set **source** to shadow rgb in one, **target** to shadow rgb in the other, then press **Set RGB**. Same for midtone and highlight areas.

Making Hue, Saturation, and Value Adjustments

For certain color correction tasks like spill suppression, you ideally want to influence only a very narrow range of color values. For such tasks, it's often helpful to use effects that employ the Hue, Saturation, and Value (HSV) color model. As its name indicates, the HSV color model breaks color into three components:

- *Hue*, which refers to the color's location on the traditional color wheel.
- *Saturation*, which refers to the extent to which the color has "soaked up" its hue.

- *Value*, which refers to the brightness of the color.

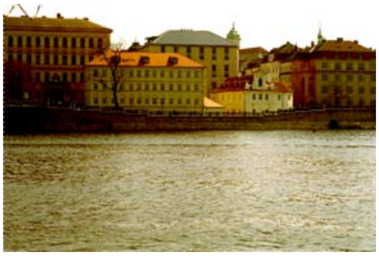


Figure 4.17: Original.

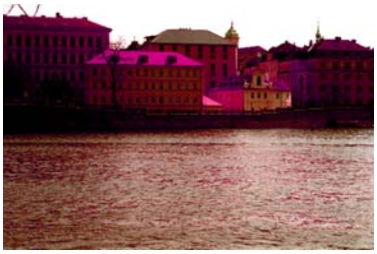


Figure 4.18: Hue shift.

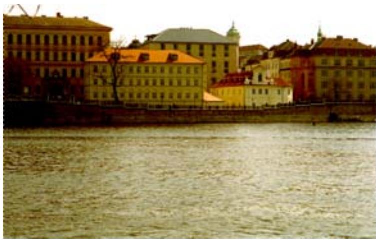


Figure 4.19: Original.

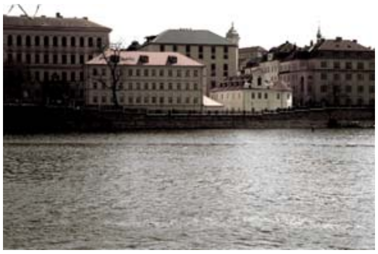


Figure 4.20: Saturation decrease.

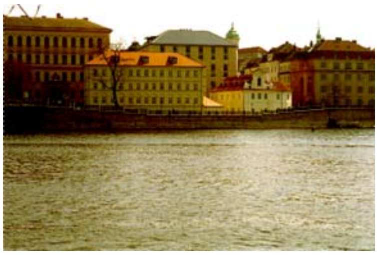


Figure 4.21: Original.

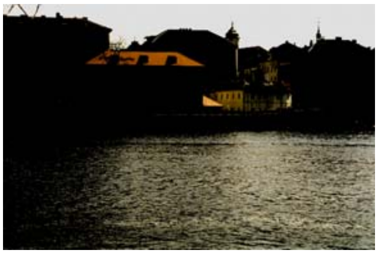


Figure 4.22: Value decrease.

Nuke offers effects that allow you to correct the hue, saturation, and value components individually or collectively.

Correcting HSV

Nuke's HSVTool node lets you simultaneously adjust hue, saturation, and value components from a single properties panel. It also features a color replacement tool. The main strength of this node is the precision it offers in limiting corrections to a narrow swath of colors.

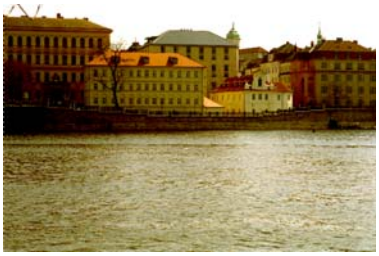


Figure 4.23: Adjusting color within a specific range of pixel values.

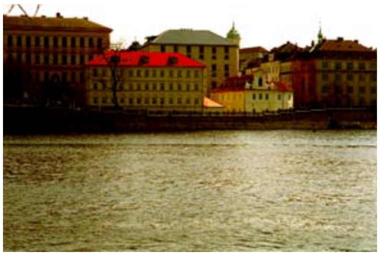


Figure 4.24: Adjusting color within a specific range of pixel values.

For example, suppose you wanted to add a bit more punch to the waterfront scene by diversifying the rooftop hues. To do so, you could limit the correction to the rooftop's ochre-colored hues by sampling a few pixels, then shift their values. Because you limited the color range, the surrounding image would be generally unaffected by the shift.

To make HSV corrections with the HSVTool node:

1. Click **Color** > **HSVTool** to insert an HSVTool node at the appropriate place in your script.
2. Connect a Viewer to the output of the HSVTool node so you can see the effect of your changes.
3. Limit, as appropriate, the range of colors you want subsequent corrections to influence:
    - In the HSVTool properties panel, click the **srccolor** color swatch. **Ctrl/Cmd**+click on the Viewer to sample a single color displayed, or **Ctrl/Cmd**+**Shift**+drag to sample a range of colors. To sample a single color from the node's input while viewing its output, **Ctrl/Cmd**+**Alt**+click on the Viewer. To sample a region from the input, **Ctrl/Cmd**+**Alt**+**Shift**+drag on the Viewer.
    - The **Range** sliders on **Hue**, **Saturation**, and **Brightness** clamp to the sampled range.
    - For any color component, drag on the **Range** sliders to expand the color range as necessary.
    - For any color component, drag on the **Range Rolloff** slider to fine tune the color range. Doing so, adjusts the amount of falloff allowed past the limits defined by the Range sliders.

4. Make the necessary HSV corrections:
    - For hue corrections, drag on the **Rotation** slider to input color wheel value between 0 and 360.
    - For saturation corrections, drag on the **Saturation Adjustment** slider to input values between -1 (completely desaturated to some shade of gray) and 1 (completely saturated).
    - For value corrections, drag on the **Brightness Adjustment** slider to input values between -1 (black) and 1 (white).

You can also make color replacements using the srccolor and dstcolor parameters: First sample the color you wish to replace with the srccolor color swath, then sample the color which you wish to use as the replacement with the dstcolor color swath. The color in dstcolor replaces the color in srccolor throughout the image.

Also, keep in mind that the HSVTool node makes an excellent keyer. You can use its Hue, Saturation, and Brightness range sliders to precisely select a range of colors, then use the channel output pulldown at the bottom of the dialog to output this selection as a matte channel. This pulldown lets you specify which color components (hue, saturation, value, etc.) are added to the matte.

Correcting Hue

Nuke's HueCorrect node lets you make precision adjustments to the levels of saturation in a range of hues. You do so via edits to a series of suppression curves.

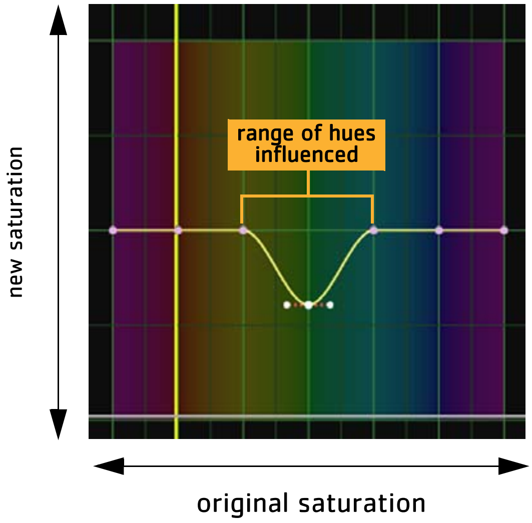


Figure 4.25: Editing the suppression curve.

By choosing which curve you edit and how much of that curve you alter, you can precisely limit the influence of the effect.

Suppressing spill

For the compositor, HueCorrect is obviously of greatest use in diminishing green, blue, or redscreen spill.

To suppress spill with the HueCorrect node:

1. Click **Color** > **HueCorrect** to insert a node at the appropriate place in your script.
2. Connect a Viewer to the output of the HueCorrect node so you can see the effect of your changes.
3. In the HueCorrect properties panel, choose the channels you want to influence:
   - Click **sat** to influence all channels (red, green, blue, and alpha) equally.
   - Click **lum** to influence all channels, but with luminance weighting in effect (meaning that the red channel receives approximately 30% of the effect; the green, 60%; and the blue, 10%).
   - Click **red** to apply the curve as a lookup on the red channel only, looking up the pixel's hue on the curve and then multiplying the red value in the pixel by the lookup result.
   - Click **green** to apply the curve as a lookup on the green channel only, looking up the pixel's hue on the curve and then multiplying the green value in the pixel by the lookup result.
   - Click **blue** to apply the curve as a lookup on the blue channel only, looking up the pixel's hue on the curve and then multiplying the blue value in the pixel by the lookup result.
   - Click **r_sup** to apply a suppression function to reduce the level of the red channel. While the red curve is used to directly multiply the red channel by the curve value, the r_sup curve is used to control the amount that the red channel is suppressed.
   - Click **g_sup** to apply a suppression function to reduce the level of the green channel. While the green curve is used to directly multiply the green channel by the curve value, the g_sup curve is used to control the amount that the green channel is suppressed.
   - Click **b_sup** apply a suppression function to reduce the level of the blue channel. While the blue curve is used to directly multiply the blue channel by the curve value, the b_sup curve is used to control the amount that the blue channel is suppressed.

Note that you can select multiple curves in order to edit one curve with reference to another.

4. If necessary, drag the cursor over the Viewer to sample the image pixels that are representative of the part of the image you want to correct. Then, in the HueCorrect properties panel, press **Ctrl+Alt** (**Cmd+Alt** on a Mac) while clicking on the curve to plot a particular pixel's value on the curve. This lets you see what portion of the curve you want to edit.
5. Edit the curve as necessary—typically this means dragging down on control points in the hue region that you wish to suppress.

Correcting Saturation

For the times when you just want to correct the saturation component and don't require limiting the correction to any particular channel, you can use Nuke's Saturation node. Its controls are bare bones—basically, just a **saturation** slider.

To make saturation corrections with the Saturation node:

1. Click **Color** > **Saturation** to insert a Saturation node at the appropriate place in your script.
2. Connect a Viewer to the output of the Saturation node so you can see the effect of your changes.
3. Drag the **saturation** slider to make the necessary corrections.

Masking Color Corrections

Virtually all the color-correction effects in Nuke include **mask** parameters that lets you limit the correction to the non-black pixel values of a matte image. For example, suppose you want to add a blue cast to the following scene without affecting the buildings.

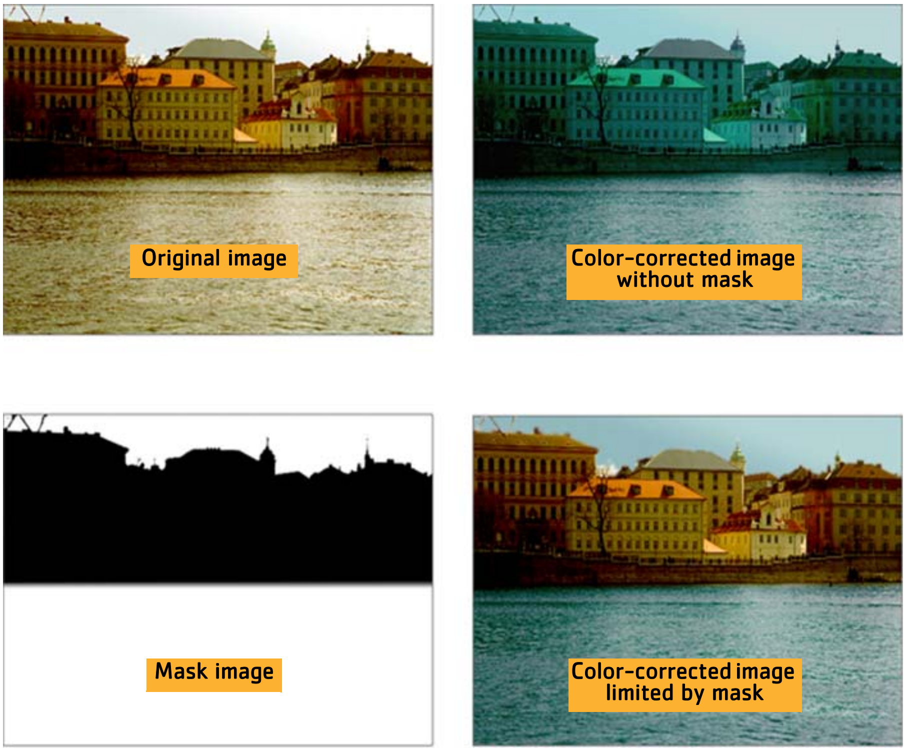
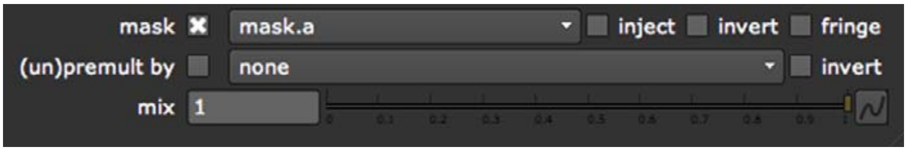

Original image
Color-corrected image without mask
Mask image
Color-corrected image limited by mask

Figure 4.26: Masking color-correction operations.

You could create a garbage mask that covers the river, then boost the red channel's gamma in the area of the frame that underlies the mask.

Typically, mask controls are located toward the bottom of the properties panel. However, in the case of multi-purpose effects like HSVTool, there may be multiple mask controls, so that you can limit each type of correction with a different mask.



mask ☒ mask.a ☐ inject ☐ invert ☐ fringe
(un)premult by ☐ none ☐ invert
mix 1

Figure 4.27: Selecting a mask channel.

To mask a color correction:

1. Open the node's properties panel and locate the **mask** controls.
2. Select the channel you wish to use as the mask from the pulldown list.

3. If you check **inject** in the **mask** controls, the mask from the **mask** input is copied into the predefined **mask.a** channel. This way, you can use the last mask input again downstream. You can also set a stream of nodes to use **mask.a** as the mask, and then change the masking of all of them by simply connecting a new mask into the mask input connector of the first node.
4. If necessary, check the **invert** box to reverse the mask.
5. To blur the edges of the mask, check **fringe**.
6. If the overall effect of the node is too harsh, you can blend back in some of the input image by dragging on the **mix** slider.
7. If you want to output only the portion of the frame underlying the mask, check the **(un)premult by** box.

Applying Grain

Grain matching—ensuring that all of the elements in a composite, including those which were digitally generated, look like they were shot on the same film stock—is often one of the final steps in achieving a convincing integration of all of a composite's elements. Nuke offers effects for synthetically creating grain and for reading in practically-created grain (grain derived from actual film stock).

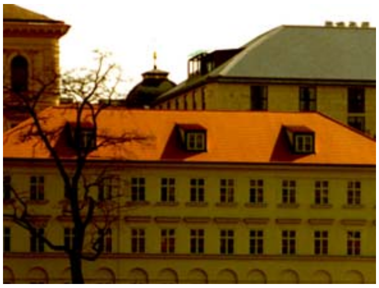


Figure 4.28: An example of applying grain to an image: Grainless image.

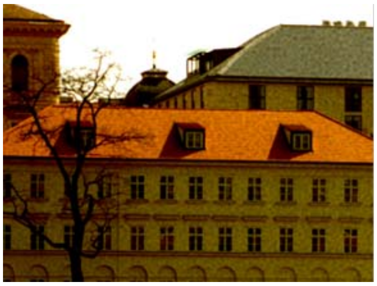


Figure 4.29: An example of applying grain to an image: Grained image.

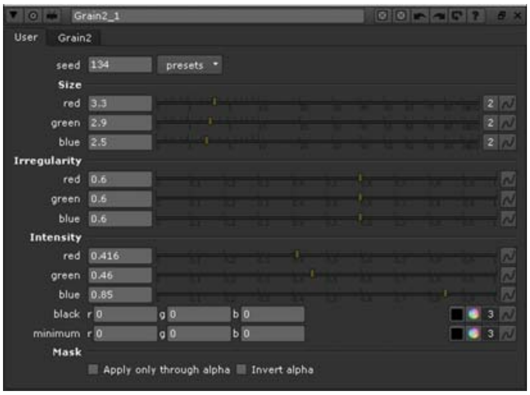
Using Synthetic Grain

Nuke offers several nodes for creating synthetic grain: Dither, Grain, and ScannedGrain. Of these, Dither is the crudest—it basically just lets you specify the amount of noise per channel.

Grain includes presets for matching film stock and a means for controlling the mix between the generated grain and the backplate. ScannedGrain offers film stock presets, plus synthetic grain controls for applying practical grain.

To add synthetic grain with the Grain node:

1. Click **Draw** > **Grain** to insert a Grain node at the appropriate place in your script.
2. Connect a Viewer to the output of the Grain node so you can see the effect of your changes.



seed 134
Size
red 3.3
green 2.9
blue 2.5
Irregularity
red 0.6
green 0.6
blue 0.6
Intensity
red 0.416
green 0.46
blue 0.85
black r 0 g 0 b 0
minimum r 0 g 0 b 0
Mask
☐ Apply only through alpha ☐ Invert alpha

3. From the **presets** pulldown menu, choose one of the film stock you want to match.
4. Adjust the **Size** sliders for the red, green, and blue channels to shrink or enlarge the granules.
5. Adjust the **Irregularity** sliders to increase or decrease the random quality of the grain, according to the different channels.
6. Adjust the **Intensity** sliders to increase or decrease the contrast of the grain against the original image.

Using Practical Grain

Although Nuke's ScannedGrain node offers controls for creating synthetic grain (ones comparable to those just discussed), it's main use is for reading in and applying scanned grain—that is, grain derived from actual film stock. If your facility has such sequences available, you can read them in and apply them using the ScannedGrain node. You can also download grain files from our website for this purpose. Both creating and downloading grain files are described below, as well as using the resulting grain files with the ScannedGrain node.

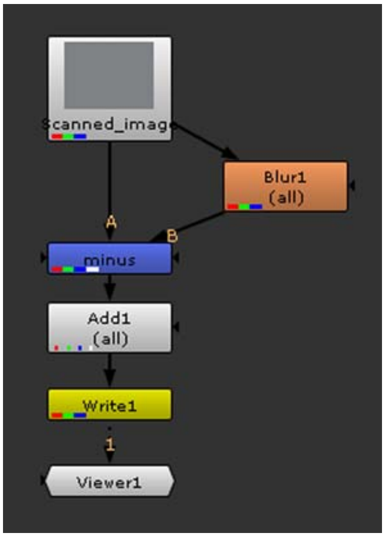
To create film stock sequences:

1. Film a gray card. Only about 50 frames are needed.
2. Scan the film in.
3. Select **Image** > **Read** to load the scanned image into Nuke.

4. Add a Blur node (**Filter** > **Blur**) after the image to blur the image until you cannot see any grain. Then, blur the image a bit more.
5. Select **Merge** > **Merge** to insert a Merge node in your script. Connect the A input of the Merge node into the original image, and the B input into the Blur node. Then, open the Merge controls and select **minus** from the **operation** pulldown menu. The blurred image is subtracted from the original image.

   The purpose of this and the previous step is to subtract any overall gray level from the grain so that only the grain is left.
6. Select **Color** > **Math** > **Add** to insert an Add node after the minus node. In the Add node controls, enter 0.5 in the **value** field. This adds a value of 0.5 to all channels.

   This step is necessary, because the ScannedGrain node subtracts 0.5 from the channels when it reads the grain file (the subtraction is needed to store negative numbers in most file formats).
7. Select **Image** > **Write** to insert a Write node after the Add node. Render the output. Any file format will do (for example, we have used the .rgb extension in the grain files on our website).

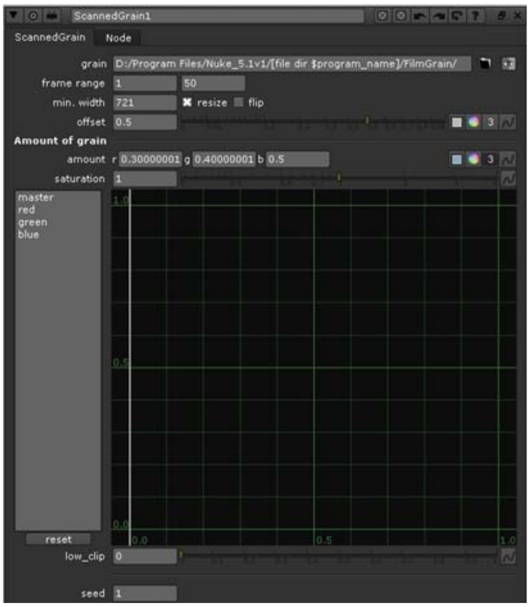


To download film stock sequences:

1. Select **Help** > **Tutorials**.
2. Click on a grain sample to download it. The downloads are in compressed tar format (tgz). The grain samples are .rgb files.

To add scanned grain with the ScannedGrain node:

1. Click **Draw** > **ScannedGrain** to insert a ScannedGrain node at the appropriate place in your script.
2. Connect a Viewer to the output of the ScannedGrain node so you can see the effect of your changes.



ScannedGrain1

grain D:/Program Files/Nuke_5.1v1/[file dir $program_name]/FilmGrain/
frame range 1 50
min. width 721 ☒ resize ☐ flip
offset 0.5
Amount of grain
amount r 0.30000001 g 0.40000001 b 0.5
saturation 1
low_clip 0
seed 1

3. Click the folder icon of the **grain** field and navigate to the appropriate film stock sequence. Select **Open**.
4. If necessary, check the **resize** box to scale the grain sequence up or down to match your working resolution.
5. In the **min. width** field, define a minimum width (in pixels) that images have to have in order to receive grain.
6. Enter values into the red, green, and blue **amount** fields to increase or decrease on a per-channel basis the density of granules. (This is accomplished, crudely speaking, by boosting or reducing the gain of the grain sequence.)

Now you're ready to fine-tune the blending between the grain and backplate.

To mix the grain and backplate:

1. Drag on the **saturation** slider to increase or decrease the intensity of the grain's hue across all channels.

2. If necessary, you can also use the supplied curve editor to edit the grain sequence's color curves. In this manner, you can alter gain, gamma, contrast, etc. on a per channel basis. (These curves function in the same manner as those describe in "Using Color Curves" on page 206).
3. To set a low threshold, based on the input image, below which the grain will not be subtracted, adjust the **low_clip** slider.

Applying Mathematical Operations to Channels

Nuke's **Color** icon in the toolbar houses a number of nodes which are designed to apply common mathematical operations to channels. These operations include clamps, offsets, inversions, multiplications, and expressions.

Clamping Channel Values

To *clamp* a channel's values is to ensure that its blackest blacks and whitest whites will be visible on an intended display device. Nuke's Clamp node lets you assign "legal" values to colors that are either too light or dark for the intended display device.

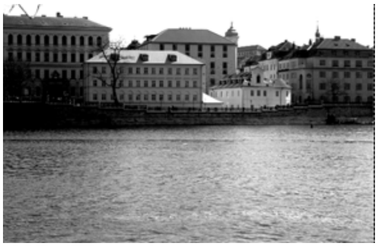


Figure 4.30: Clamping black and white pixels to "legal" values.

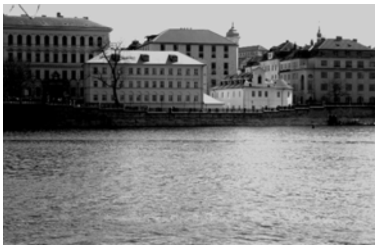


Figure 4.31: Clamping black and white pixels to "legal" values.

For this effect, you use Nuke's Clamp node.

To clamp channel values:

1. Click **Color** > **Clamp** to insert a Clamp node at the appropriate point in your script.
2. Connect a Viewer to the output of the Clamp node so you can see the effect of your changes.
3. In the Clamp properties panel, use the **channels** field to select the channel you wish to clamp.
4. Drag the **minimum** slider to the legal value. (This has the effect of causing black values to go gray.)
5. Drag the **maximum** slider to the legal value. (This has the effect of causing white values to go gray.)

Offsetting Channel Values

To *offset* a channel's values is to add a fixed value to them, which, in effect lightens the whole channel. You can also add a negative value to a channel, in which case the channel gets darker.

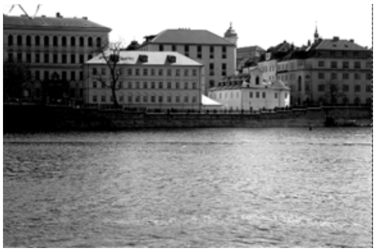


Figure 4.32: Offsetting channel values.

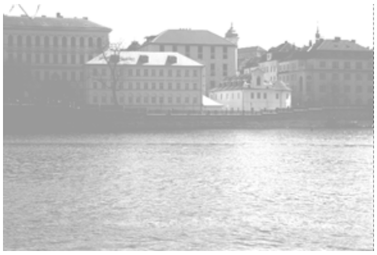


Figure 4.33: Offsetting channel values.

For this effect, you use Nuke's Add node.

To offset channel values:

1. Click **Color** > **Math** > **Add** to insert a Add node at the appropriate point in your script.
2. Connect a Viewer to the output of the Add node so you can see the effect of your changes.
3. In the Add properties panel, use the **channels** field to select the channel you wish to offset.
4. Use the **value** slider to input the value you wish to add to the channel's values.
5. If you are using premultiplied input images, you may want to check **(un)premult by** and select **rgba.alpha** from the pulldown menu. This will simulate doing the addition before the premultiplication was done.

Inverting Channel Values

To *invert* a channel is to subtract its values from one, which causes its blacks to become white and its whites to become black. In the course of building a script, you'll have frequent need to invert mattes in particular.

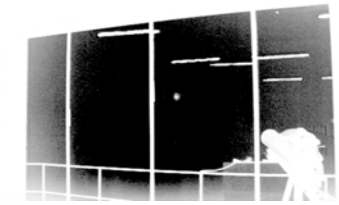


Figure 4.34: Inverting channel values.

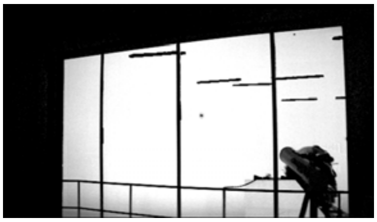


Figure 4.35: Inverting channel values.

To invert channels you use Nuke's Invert node.

To invert channel values:

1. Click **Color** > **Invert** to insert an Invert node at the appropriate point in your script.
2. Connect a Viewer to the output of the Invert node so you can see the effect of your changes.
3. In the Invert properties panel, use the **channels** field to select the channel you wish to invert.

Multiplying Channel Values

To *multiply* a channel's values is to times them by a given factor, which has the effect of lightening the channel while preserving the blackpoint. (This operation is also knows as gain.)

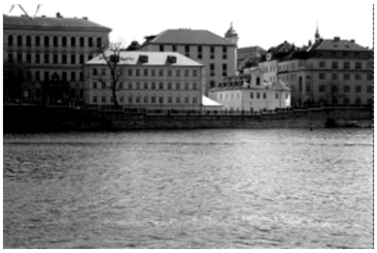


Figure 4.36: Multiplying channel values.

Figure 4.37: Multiplying channel values.

For this effect, you use Nuke's Multiply node.

To multiply channel values:

1. Click **Color** > **Math** > **Multiply** to insert a Multiply node at the appropriate point in your script.
2. Connect a Viewer to the output of the Multiply node so you can see the effect of your changes.
3. In the Multiply properties panel, use the **channels** field to select the channel whose values you wish to multiply.
4. Use the **value** slider to input the factor by which to you want to times the channel's values.

Applying Expressions to Channel Values

Up till now, the discussion has focused on how to apply simple mathematical formulae—additions, subtractions, multiplications, etc.—to a channel's values. Nuke's Expression node, however allows you to apply complex formulae to a channel's values. The actual syntax for expressions is rather complex, and thus must be deferred to Chapter 18, *Expressions,* on page 533. For now, you can read about the basics of how to operate the Expression node.

To apply expressions to channel values:

1. Click **Color** > **Math** > **Expression** to insert an Expression node at the appropriate point in your script.
2. Connect a Viewer to the output of the Expression node so you can see the effect of your changes.
3. In the Expression properties panel, use the channel menus and buttons to select the channel to which you wish to apply an expression.
4. Type the actual expression in the = field next to the channel.

   For example, to assign noise to the red channel, then boost the gain of that result by 20 you would type **(random*r)*20**.
5. If necessary, you can apply different expressions to different sets of channels by repeating the above steps for the other channel menus and buttons.
6. If you need to use a long expression in several fields, you can use the fields on top of the properties panel for assigning the expression temporarily to a variable. Enter your variable on the left side of the = sign, and the expression on the right. You can then use the variable to represent the expression in the = fields next to the channels.

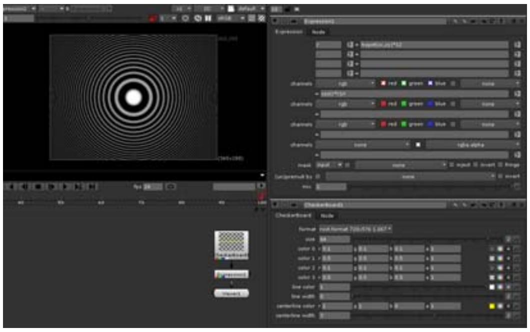


Figure 4.38: A checkerboard modified using an Expression node.

Transforming the Color Space

Whenever you read a clip into a script, it is automatically converted to Nuke's native color space, which is 32-bit per channel RGB, a linear format. This conversion takes place even if the clip you read in is in the Kodak Cineon format, which is a logarithmic format.

The reverse of this conversion, called a lin-to-log conversion, also automatically takes place when you write the processed element back out of the script—that is, Nuke automatically converts it back into a Cineon element.

Overriding the Default Cineon Conversion

Nuke uses the Kodak-recommended settings when making Cineon conversions in either direction. It's rare that you would want to override these settings, but if it becomes necessary you can use Nuke's log2lin node.

To override the default Cineon conversions:

1. Double-click on the Read node of the Cineon element whose conversion you wish to override.
2. In the Read properties panel, set the **colorspace** pulldown list to **linear**. This halts the automatic log-to-lin conversion.
3. Click **Color** > **Log2Lin** to insert a log2lin node directly after the Read node.
4. In the log2lin properties panel, set the **operation** pulldown to **log2lin**.
5. Set **black**, **white**, and **gamma** to the appropriate values.
6. Copy the log2lin node and insert it just before the element's Write node.
7. Open up the properties panel of the second log2lin node and set the **operation** pulldown list to **lin2log**. This gives you the reverse of the conversion you created above.
8. Double click on the element's Write node.
9. In the Write properties panel, set the **colorspace** pulldown list to **linear**. This halts the automatic lin-to-log conversion and lets the one you create above have priority.

Making Other Color Space Conversions

You can also convert elements from Nuke's native color space to other color spaces more appropriate to a given process or intended display device. For conversions such as this, use Nuke's Colorspace node, which supports RGB, HSV, YUV, CIE, and CMS formats (and various subformats).

To convert an element in Nuke's native color space into another color space:

1. Click **Color** > **Colorspace** to insert a Colorspace node into the appropriate place in your script.
2. In the Colorspace properties panel, set the rightmost pulldown menu in the **out** controls to the appropriate standard.

3. Set the pulldown menu in the middle of the **out** controls to the appropriate standard.
4. Set the leftmost pulldown menu in the **out** controls to the color space of your choice.
5. If you wish to reverse this conversion later in the script:
   - Copy the Colorspace node and insert it at the appropriate point in your script.
   - Set the **out** controls to **sRGB**, **D55**, and **RGB**.
   - Set the **in** controls to match the values you entered in steps 2, 3, and 4 above.
6. If you wish write out the element in the new color space:
   - Double click on the element's Write node.
   - In the Write properties panel, set the **colorspace** pulldown list to **linear**. This halts the automatic conversion and lets the one you create above have priority.

Changing the Viewer Color Space

By default, a script's Viewers display images in Nuke's native color space. You can, however, set a script's Viewers to display images in non-native color spaces. Changing the display color space in no way affects your rendered output. You are applying a display-only lookup table.

To change the displayed color space for individual Viewers:

Select the desired color space from the Viewer's Viewer process menu on the top right corner. For more information on this menu, see "Input Process and Viewer Process controls" on page 103.

5 TRANSFORMING ELEMENTS

This chapter explains how to perform a range of 2D and 2.5D spatial transformations. You learn how to apply geometric transformations (including translations, rotations, scales, and skews) to elements, and how to add motion blur using the nodes in the Transform menu.

Note that this chapter discusses how to manually apply transformations. Chapter 6, "Tracking and Stabilizing" discusses how to use Nuke's tracker to automatically generate and apply transformations.

Transforming in 2D

This section describes how to apply 2D transformations including translations, rotations, scales, and skews to elements using a number of Nuke nodes.

Using the 2D Transformation Overlay

Several of the nodes discussed in this section display a Viewer overlay for executing spatial transformations. This overlay is often a faster alternative to the properties panel. The figure below shows you how to use Nuke 2D transformation overlay.

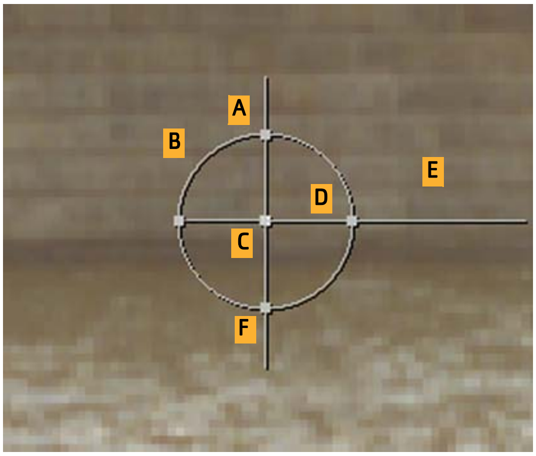


Figure 5.1: Transformation Overlay.

A. Drag to skew the frame (see "Skewing Elements" on page 235).

B. Drag to scale the frame uniformly—simultaneously on x and y (see "Scaling Elements" on page 239).

C. Drag to translate the frame (see "Translating Elements" on page 238).

Shift+drag to constrain the translation to x or y.

Ctrl/**Cmd**+drag to reposition the *pivot point* (the point that acts as the center to transformation operations).

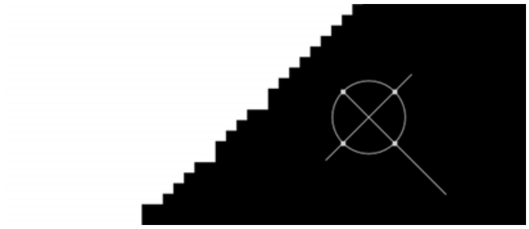
D. Drag to scale the frame on x.

E. Drag to rotate the frame around the pivot point (see "Rotating Elements" on page 238). The transform overlay snaps to typical values. To prevent the snapping, press **Shift** while dragging.

F. Drag to scale the frame on y.

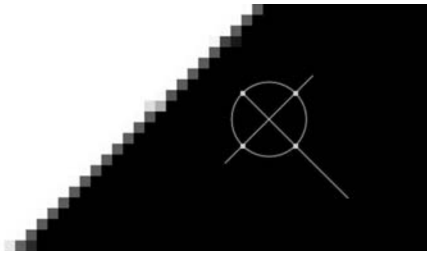
Choosing a Filtering Algorithm

Spatial transformations involve remapping pixels from their original positions to new positions. The question arises as to what values to assign remapped pixels. In the simplest case, they retain their original values, but this can create problems with image quality, particularly in high contrast areas of the frame. For example, the figure below shows a close up a high-contrast feature that has been rotated clockwise by 45 degrees. The remapped pixels have retained their original values, but the result is a highly aliased, or jaggy, edge:



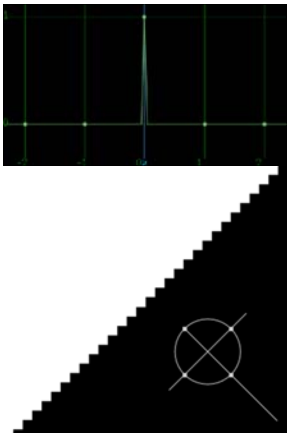
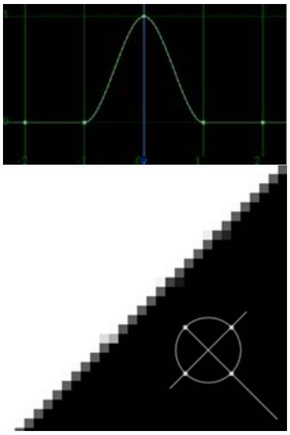
The solution is to apply a more sophisticated *filtering algorithm* to determine the values of remapped pixels—one that takes into account, in some fashion, the values of neighbouring pixels.

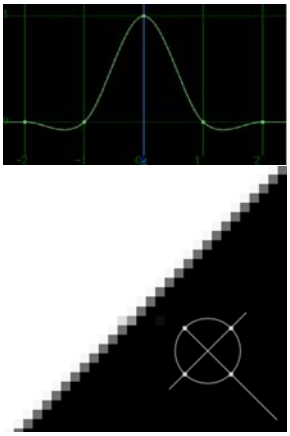
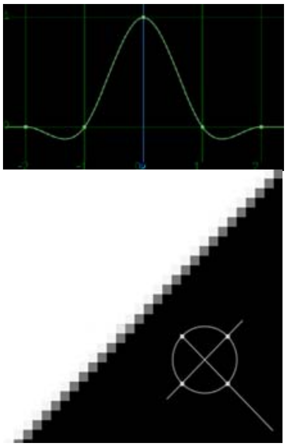
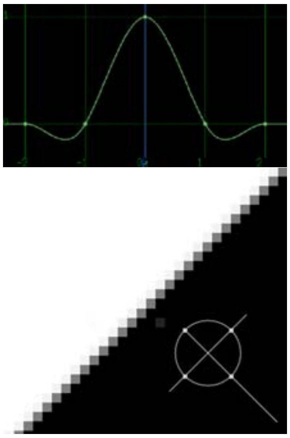
For example, applying Nuke's cubic algorithm to the above rotation, results in a softer, less jagged edge:

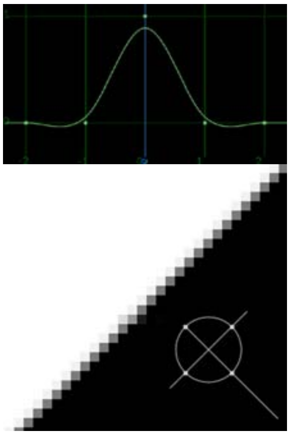
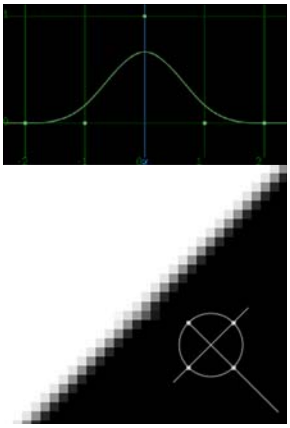
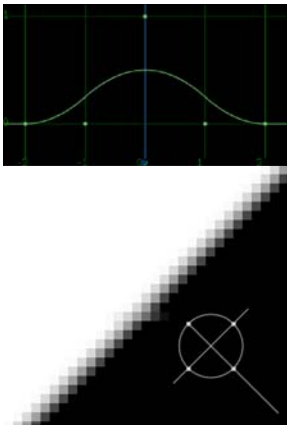


When executing spatial transformations, Nuke lets you choose from the filtering algorithms described in the table below.

Note that the curves shown in the table plot the manner by which each algorithm samples from neighbouring pixels. The center of each curve represents the value of the remapped pixel itself, and the rising and falling portions of each curve represent the amount of sampling that occurs across a five pixel radius.

Filter	Description	Sampling Curve and Output
Impulse	Remapped pixels carry original values.	
Cubic (default)	Remapped pixels receive some smoothing.	

Filter	Description	Sampling Curve and Output
Keys	Remapped pixels receive some smoothing, plus minor sharpening (as shown by the negative -y portions of the curve).	
Simon	Remapped pixels receive some smoothing, plus medium sharpening (as shown by the negative -y portions of the curve).	
Rifman	Remapped pixels receive some smoothing, plus significant sharpening (as shown by the negative -y portions of the curve).	

Filter	Description	Sampling Curve and Output
Mitchell	Remapped pixels receive some smoothing, plus blurring to hide pixelation.	
Parzen	Remapped pixels receive the greatest smoothing of all filters.	
Notch	Remapped pixels receive flat smoothing (which tends to hide *moire* patterns).	

How Your Nodes Concatenate

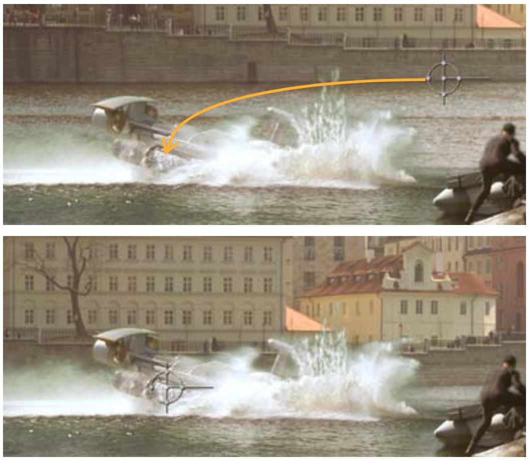
Concatenation is behavior that some Nuke nodes perform when you have several nodes transforming or color correcting your image one after another. When nodes concatenate, they pass on these adjacent transformation operations to the last transforming node in the row and the last node then performs all the transformations at once. This improves the picture quality because the pixels only get remapped once.

In order to concatenate, the concatenating nodes have to be adjacent. So, if you have a node that doesn't concatenate (a Crop node for example) between two concatenating nodes (for example Transform nodes), no concatenation will occur. The nodes also need to perform similar operations in order to concatenate with each other, for example transform nodes only concatenate with other transform nodes. As a rule of thumb, nodes that concatenate are usually either color correction nodes or transform nodes.

If you're using more than one filtering method in the nodes that concatenate, the last filtering method in the series of concatenating nodes will be applied on the result.

Translating Elements

To *translate* an element is to slide it on x or y.



You can use the Transform, TransformMasked, or Position nodes to translate elements.

Using the Transform node

The Transform and TransformMasked nodes let you not only translate elements, but also rotate, scale, and skew them from a single properties panel.

TransformMasked is identical to Transform except that it offers controls for assigning a mask to protect certain areas of the frame from translations. For the sake of brevity, this chapter only discusses the use of Transform, but keep in mind you can use TransformMasked any time you need to process a transformation through a mask. Its mask controls work in the same fashion as those described in "Masking Color Corrections" on page 214.

To translate an element using the Transform node:

1. Click **Transform** > **Transform** to insert a Transform node at appropriate place in your script.
2. Connect a Viewer to the output of the Transform node so you can see the effect of your changes.
3. In the Transform properties panel, increment or decrement the **translate x** and **y** fields to slide the element along either axis.

   Or drag on the center of the transformation overlay.

Using the Position node

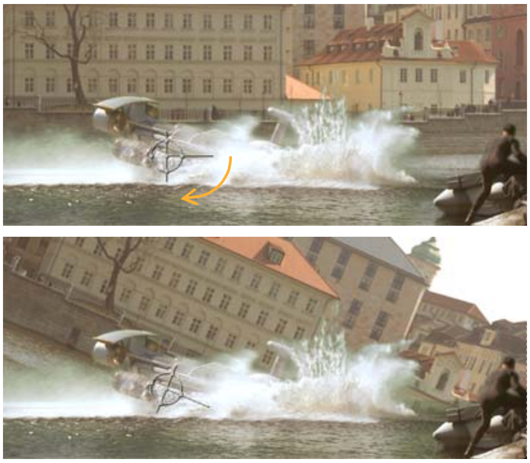
The Position node gives you just bare-bones parameters for translating an element.

To translate an element using the Position node:

1. Click **Transform** > **Position** to insert a Position node at appropriate place in your script.
2. Connect a Viewer to the output of the Position node so you can see the effect of your changes.
3. In the Position properties panel, increment or decrement the **translate x** and **y** fields to slide the element along either axis.

Rotating Elements

To *rotate* an element is to spin it around the pivot point.



Use the Transform node to rotate elements.

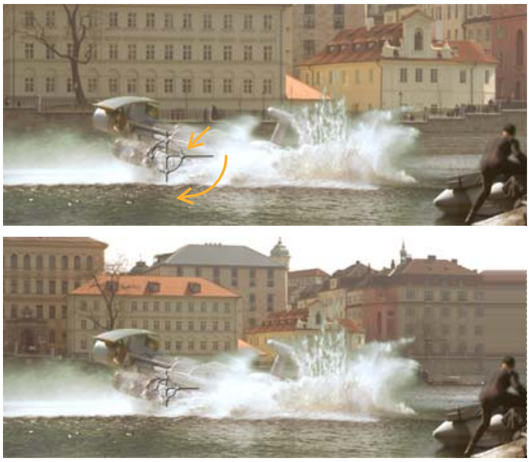
To rotate an element using the Transform node:

1. Click **Transform** > **Transform** to insert a Transform node at appropriate place in your script.
2. Connect a Viewer to the output of the Transform node so you can see the effect of your changes.
3. In the Transform properties panel, choose the appropriate filtering algorithm from the **filter** pulldown list (see "Choosing a Filtering Algorithm" on page 227).
4. Position the pivot point as necessary:
    - Increment or decrement the **center x** and **y** fields to move the axis in either direction.
    - Or press **Ctrl** (**Cmd** on a Mac) while dragging on the center of the transformation overlay.
5. Increment or decrement the **rotate** field.

   Or drag on the horizontal bar of the transformation overlay.

Scaling Elements

To *scale* an element is to resize it by adding (upsampling) or removing (downsampling) pixels.



Nuke offers several nodes for scaling elements. Transform, whose scaling functions are described below, is designed primarily for scaling up or down the background plate in a composite.

Reformat is designed for writing out elements with specific resolutions and pixel aspect ratios. "Adding Motion Blur" on page 240 describes the use of this node.

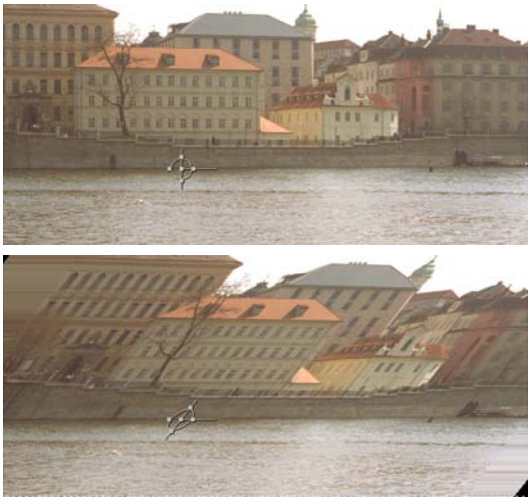
To scale an element using the Transform node:

1. Click **Transform** > **Transform** to insert a Transform node at appropriate place in your script.
2. Connect a Viewer to the output of the Transform node so you can see the effect of your changes.
3. In the Transform properties panel, choose the appropriate filtering algorithm from the **filter** pulldown list (see "Choosing a Filtering Algorithm" on page 227).
4. Position the pivot point as necessary:
   - Increment or decrement the **center x** and **y** fields to move the axis in either direction.
   - Or press **Ctrl** (**Cmd** on a Mac) while dragging on the center of the transformation overlay.
5. To scale the frame uniformly (on both x and y):
   - Increment or decrement the Transform node's **scale** field.
   - Or drag the circle-portion of the of the transformation overlay.
6. To scale the frame asymmetrically (on x or y):
   - Click **scale** parameter's channel chooser to reveal the **x** and **y** fields, then increment or decrement each individually.

  - Or drag any of the four points on the circle-portion of the transformation overlay. The top and bottom points scale on y; the left and right points, on x.

Skewing Elements

To *skew* an element is to rotate its pixel columns around the pivot point.



Use the Transform node to skew elements.

To skew an element using the Transform node:

1. Click **Transform** > **Transform** to insert a Transform node at appropriate place in your script.
2. Connect a Viewer to the output of the Transform node so you can see the effect of your changes.
3. In the Transform properties panel, choose the appropriate filtering algorithm from the **filter** pulldown list (see "Choosing a Filtering Algorithm" on page 227).
4. Position the pivot point as necessary:
   - Increment or decrement the **center x** and **y** fields to move the axis in either direction.
   - Or **Ctrl**+drag (**Cmd**+drag on a Mac) on the center of the transformation overlay.
5. Increment or decrement the **skew** field to rotate the pixel columns around the pivot point.

   Or drag the vertical bar of the transformation overlay.

To invert a transform effect

You can invert the effect you've created with the Transform node by checking the **invert** box in the Transform control panel. This will use the inverse values of your translate, rotate, scale and skew values. When the box is checked, a small transform handle appears next to the standard transform handle in the Viewer.

Applying Core Transformations in 2.5D

Nuke's Card3D node lets you apply the same geometric transformations possible with the Transform node, but gives you an additional axis of operation, z.

Just to be clear, the Card3D node's transformations are not truly 3D, but rather what is sometimes called "2.5D"—meaning that you can move an element back on the z axis, but doing so does not convey the sense that it is behind or in front of another element. 2.5D transformations are useful for tasks like "cheating" the perspective of an element or "faking" a camera zoom.

Remember, however, that Nuke doesn't limit you to 2.5 dimensions. If you need true 3D capabilities, you can construct a 3D scene. See Chapter 15, *3D Compositing,* on page 426.

Adding a Card3D Node

To add a Card3D node:

1. Click **Transform** > **Card3D** to insert a Card3D node at appropriate place in your script.
2. Connect a Viewer to the output of the Card3D node so you can see the effect of your changes.

Specifying the Order of Operations

The order by which Nuke executes operations can affect the outcome. The Card3D node lets you choose the order by which Nuke executes scales, rotations, and translations, as well as the order by which it executes rotation on individual axes.

To choose the operation order for scales, rotations, and translations:

In the Card3D properties panel, select an option from the **transform order** pulldown list, which displays all possible combinations (**S** signifies scale, **R**, rotation; and **T**, translation).

To choose the operation order for rotations:

Select an option from the **rotation order** pulldown list, which displays all possible axial combinations.

Choosing a Filtering Algorithm

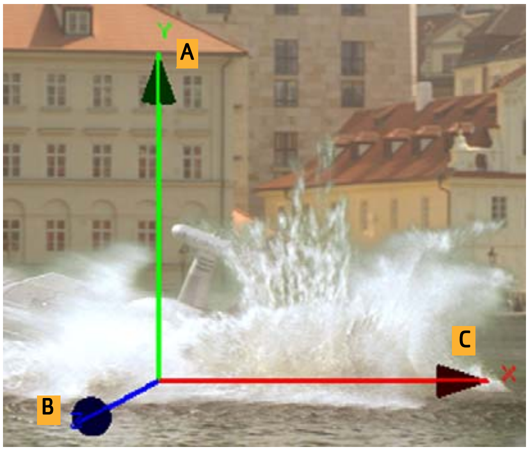
Filtering algorithms let you specify the degree of smoothing and sharpening that remapped pixels receive during transformation. The Card3D node offers the same filter algorithms as the Transform node. See "Choosing a Filtering Algorithm" on page 227 for more information.

To choose a filter algorithm:

Select the desired algorithm from the **filter** pulldown list.

Using the 3D Transformation Handles

You'll note when viewing the output of a Card3D node that it displays an overlay for executing spatial transformations. This overlay is often a faster alternative to the properties panel. The figure below shows you how to use it.



A. Drag to translate the frame on the y axis (see "Translating Elements" on page 231).

   Press **Ctrl/Cmd** while dragging to rotate the frame on any axis (see "Rotating Elements" on page 232).

B. Drag to translate the frame on the z axis.

   Press **Ctrl/Cmd** while dragging to rotate the frame on any axis.

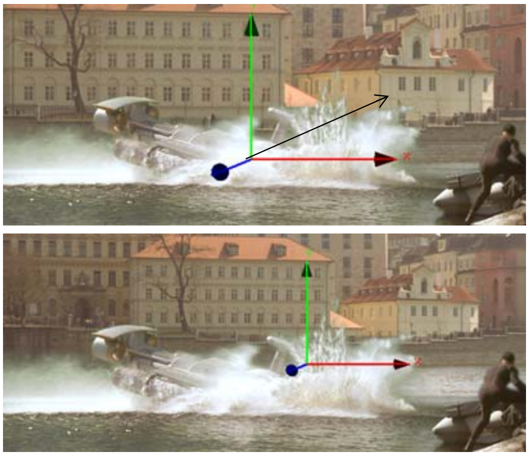
C. Drag to translate the frame on the x axis.

   Press **Shift** while dragging to constrain the translation to x.

Press **Ctrl/Cmd** while dragging to rotate the frame on any axis.

Translating Elements

When using the Card3D node, you can translate elements on z in addition to the other axes.



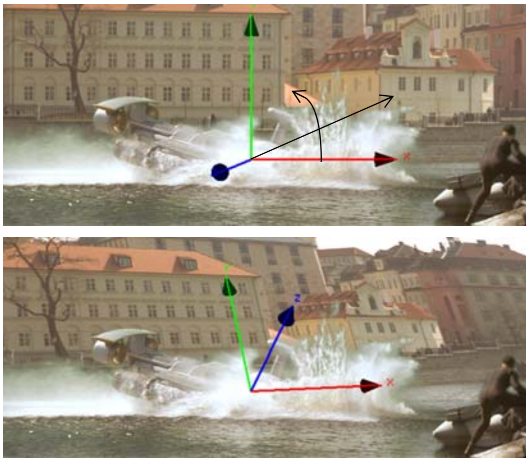
To translate an element using the Card3D node:

In the Card3D properties panel, increment or decrement the **translate x**, **y**, and **z** fields to slide the element along any axis.

Alternatively, you can drag on any axis on transformation overlay.

Rotating Elements

When using the Card3D node, you can rotate elements around the x and y axes, in addition to the z. This is useful for cheating the perspective.



To rotate elements using the Card3D node:

1. Position the pivot point as necessary by incrementing or decrementing the **pivot** **x**, **y**, and **z** fields to move the axis in any direction.

   Alternatively, you can position the pivot point by pressing **Ctrl/Cmd+Alt** while dragging.
2. Increment or decrement the **rotate** **x**, **y**, and **z** fields to spin the element around the pivot point.

   Alternatively, you can press **Ctrl/Cmd** while dragging on any axis on the transformation overlay.

Scaling Elements

To scale an element using the Card3D node:

1. Position the pivot point as necessary by incrementing or decrementing the **pivot** **x**, **y**, and **z** fields to move the axis in any direction.

   Alternatively, you can position the pivot point by pressing **Ctrl/Cmd+Alt** while dragging.
2. To scale the frame simultaneously on x, y, and z, increment or decrement the **uniform scale** field.
3. To scale the frame asymmetrically, increment or decrement the **scale** **x**, **y**, and **z** fields.

Skewing Elements

Whereas the Transform node lets you rotate pixel columns only around the z axis, Card3D permits you to so around all three axes.

To skew an element using the Card3D node:

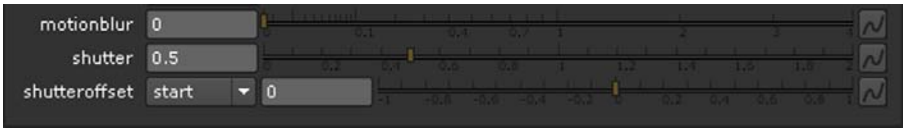
1. Position the pivot point as necessary by incrementing or decrementing the **pivot x**, **y**, and **z** fields to move the axis in any direction.

   Alternatively, you can position the pivot point by pressing **Ctrl/Cmd+Alt** while dragging.
2. Increment or decrement the **skew x**, **y**, and **z** fields to rotate the pixel columns around the corresponding axes.

Adding Motion Blur

The following nodes under the Transform menu have their own controls for adding motion blur to transformations:

- Transform
- TransformMasked
- Card (3D)
- CornerPin2D
- Reconcile3D
- Tracker
- Stabilize2D.



motionblur 0
shutter 0.5
shutteroffset start 0

These controls allow you to create motion blur without adding a separate node for it. The output is similar to a TimeBlur node (see "Applying the TimeBlur Filter" on page 382), but rather than averaging the results of several whole images computed at steps over the shutter period, a number of samples are taken at many random times over the shutter period. This effectively gives many more "steps" and thus a smoother looking result for a smaller total number of computations.

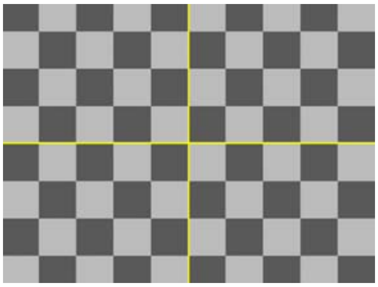


Figure 5.2: Before rotation and motion blur.

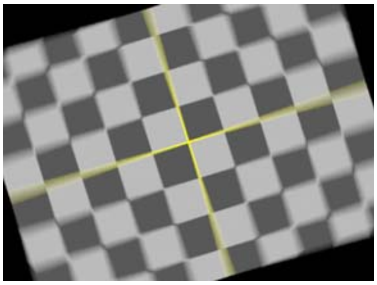
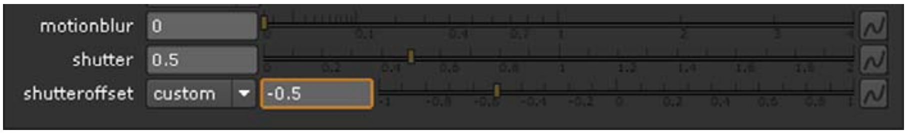


Figure 5.3: After rotation and motion blur.

When using several of these nodes in a row, the motion blur is concatenated, and the last transform in the chain defines the motion blur applied.

To add motion blur:

1. Open the transform node's controls.
2. Create a transform and animate it. For instructions on how to do this, see "Animating Parameters" on page 71.
3. In the **motionblur** field, enter the sampling rate. This affects the number of times the input is sampled over the shutter time. The higher the rate, the smoother the result. In many cases, a value of 1.0 is enough. Setting the value to 0 produces no motion blur.
4. In the **shutter** field, enter the number of frames the shutter stays open when motion blurring. For example, a value of 0.5 would correspond to half a frame. Increasing the value produces more blur, and decreasing the value less.
5. From the **shutteroffset** pulldown menu, select when the shutter opens and closes in relation to the current frame value:
   - to center the shutter around the current frame, select **centerd**. For example, if you set the **shutter** value to 1 and your current frame is 30, the shutter will stay open from frame 29,5 to 30,5.
   - to open the shutter at the current frame, select **start**. For example, if you set the **shutter** value to 1 and your current frame is 30, the shutter will stay open from frame 30 to 31.
   - to close the shutter at the current frame, select **end**. For example, if you set the **shutter** value to 1 and your current frame is 30, the shutter will stay open from frame 29 to 30.
   - to open the shutter at the time you specify, select **custom**. In the field next to the pulldown menu, enter a value (in frames) you want to add to the current frame. To open the shutter before the current frame, enter a negative value. For example, a value of -0.5 would open the shutter half a frame before the current frame.



motionblur 0
shutter 0.5
shutteroffset custom -0.5

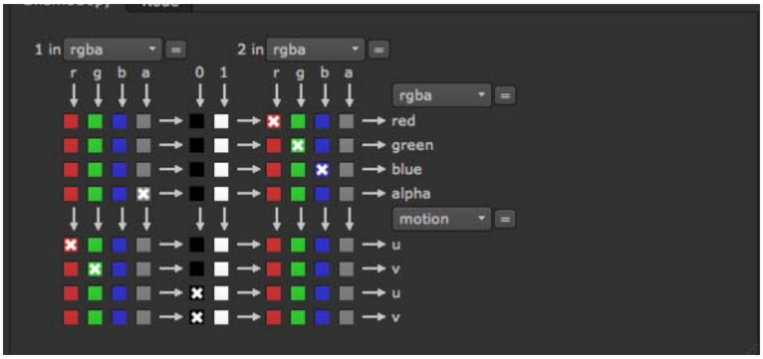
To add motion blur to an image rendered in a third-party application

Another way to add motion blur to your image is to use the VectorBlur node. VectorBlur takes each of your image's pixels and blurs them in a straight line, using the u and v channels to determine the blur direction.

VectorBlur expects the values from your input plates to be pixel space screen units, in other words one unit should equal to one pixel. Nuke uses this information to calculate the distance that one pixel travels between two frames. So, in order to get working motion blur result, you should make sure Nuke is getting correct values to work with. Particularly if you've used a third party application to create your input files, you might have files using varying values. The following is an example of creating motion blur with the VectorBlur node using files written from a third party application.

To create motion blur with the VectorBlur node:

1. Read in your footage and motion blur files, for example an EXR file with a spinning donut and a SGI file with motion blur vectors that are normalised to have values between 0 and 1.
2. Using the ShuffleCopy node, select which channels VectorBlur should read from your motion vector file (node input 1) and color image file (node input 2). In this case, you would use the motion vector file's red and green channels as the motion u and v channels, and its alpha channel as the alpha channel. Meanwhile, the image file would output the red, green and blue channels for the main color image. With this setup, your ShuffleCopy node controls would look like this



3. Connect the VectorBlur node to the ShuffleCopy node. You also need to tell VectorBlur which motion vector channels to use, so change the **uv channels** control to **motion**.

4. If your motion vectors have been normalised to be between 0 and 1, you can set the u and v values in the **add** control to **-0.5** to offset the motion blur center. This will usually be necessary for any motion vectors stored in an integer file format like 16 bit TIFF or TGA. Vectors that go to negative x or y directions use half the numbers in the range and vectors that go positive use the other half..
5. With the **multiply** and **offset** controls, you can further adjust the amount of motion blur you want to produce. The **offset** value allows you to correct for normalization of your vector values, and the **multiply** value controls the magnitude of them.
6. If the vectors have been premultiplied with the alpha channel, their value is not accurate in places where the alpha is not 1.0. You'll want to check the **alpha** checkbox to use the input image's alpha channel to help VectorBlur to deal with motion vectors that have been premultiplied by this alpha channel.
7. Getting a good, even motion blur result largely depends on choosing the right calculation method. In the method dropdown, choose:
   - backward - backward method is effective and fast but may not be accurate if you don't have motion vector values at all pixels throughout the whole frame
   - forward - the forward method is slower, but it will give you a more accurate result, especially in cases where the vectors don't cover the whole frame. In this case we know the motion vectors are not continuous, so choosing forward is a good option.

.

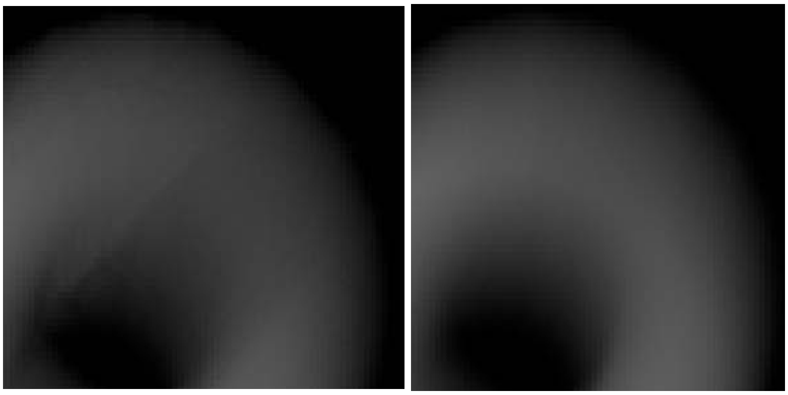


Figure 5.4: VectorBlur alpha control off (left) and on (right)

Replicating the Input Image Across the Output

The Tile node produces an output image that contains scaled-down, tiled copies of the input image. The output image is the same format as the input.

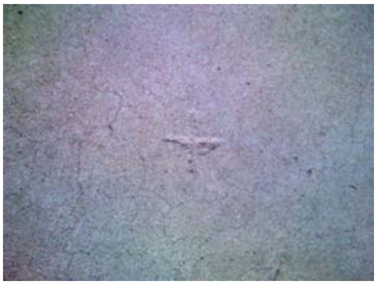


Figure 5.5: Before the Tile node.

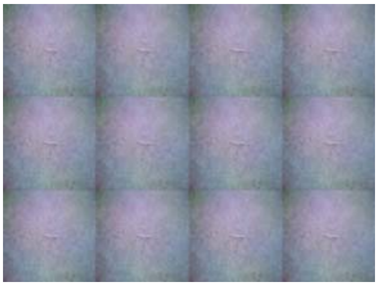
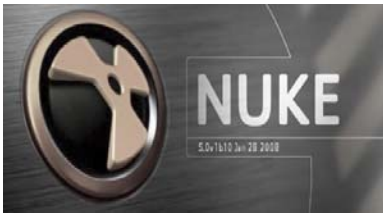


Figure 5.6: After the TIle node.

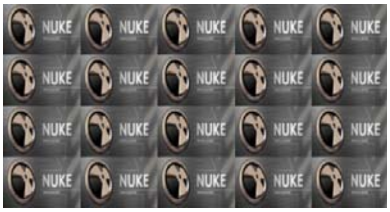
To use the Tile node:

1. Select the image you want to replicate and choose **Transform** > **Tile**.

   A Tile node is inserted in the Node Graph.
2. Attach a Viewer to the Tile node.
3. In the Tile node properties, use the **rows** field to define how many times the image is replicated vertically. Note that the value can be fractional.

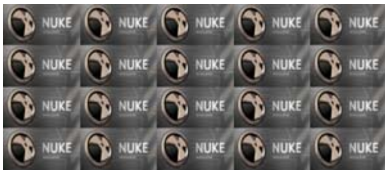


The original input image.

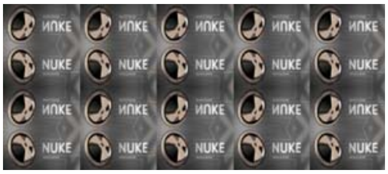


The output of the Tile node with rows set to 4 and columns to 5.

   If you want to flip adjacent tiles vertically to form mirror images, check **mirror**.



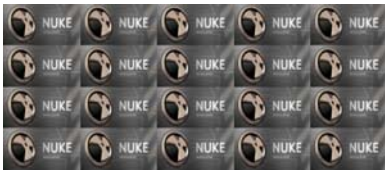
The output of the Tile node without mirroring.



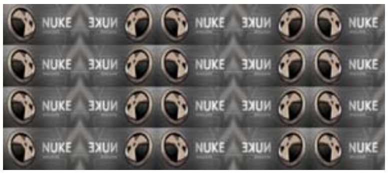
The output of the Tile node with vertical mirroring.

4. In the **columns** field, enter the number of times you want to replicate the image horizontally. Note that the value can be fractional.

If you want to flip adjacent tiles horizontally to form mirror images, check **mirror**.



The output of the Tile node without mirroring.



The output of the Tile node with horizontal mirroring.

5. From the **filter** menu, choose an appropriate filtering algorithm. For more information, see "Choosing a Filtering Algorithm" on page 227.

6 TRACKING AND STABILIZING

Nuke features a 2D tracker that allows you to extract animation data from the position, size, and rotation of an image. Using expressions, you can apply the data directly to transform and matchmove another element. Or you can invert the values of the data and apply it to the original element—again through expressions—to stabilize the image.

This is the general process for tracking an image:

1. Connect a Tracker node to the image you want to track.
2. Place tracking anchors over features in the image.
3. Calculate the tracking data.
4. Choose the tracking operation you want to perform: stabilize or matchmove.

Before you track, it's important to playback the image several times. This will help you identify the best features for the process, as well as any problems with motion blur or features moving out of frame.

For some images, you may need to filter or color-correct the image to boost the visibility of features before you attempt to track them. Because of the procedural nature of Nuke, you can disable these extra nodes after you get a successful track, or simply reconnect the Tracker node at the appropriate place to apply the transform.

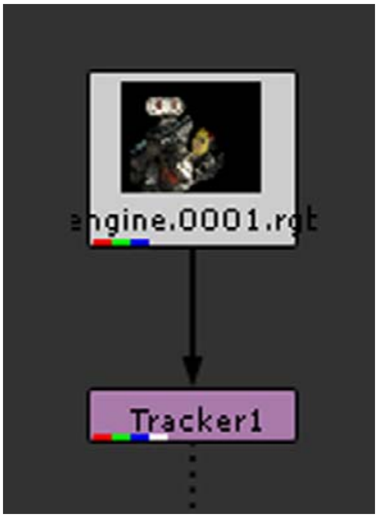
Tracking an Image

The Tracker can analyze the movement of up to four different features in a single image. Nuke generates one animation curve or *track* for each feature.

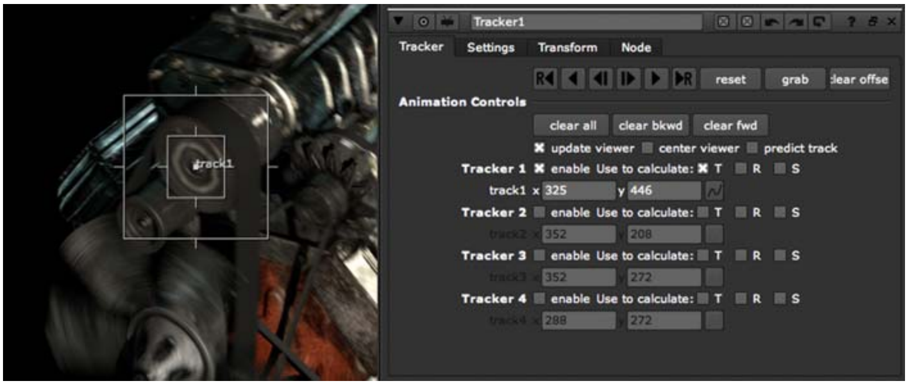
A single track is usually sufficient to record a feature's horizontal and vertical position across the 2D plane. Two or more tracks are required to extrapolate scaling and rotation.

To track position only (a single feature):

1. Select the node that outputs the image you want to track.
2. Choose **Transform > Tracker** to connect a new Tracker node.


engine.0001.rgb
Tracker1

3. Click the **Tracker** tab. In the Viewer, you will see the anchor for the first track.


track1
Tracker1
Tracker
Settings
Transform
Node
reset
grab
clear offse
Animation Controls
clear all
clear bkwd
clear fwd

☒ update viewer ☐ center viewer ☐ predict track

Tracker 1 ☒ enable Use to calculate: ☒ T ☐ R ☐ S

track1 x 325 y 446

Tracker 2 ☐ enable Use to calculate: ☐ T ☐ R ☐ S

track2 x 352 y 208

Tracker 3 ☐ enable Use to calculate: ☐ T ☐ R ☐ S

track3 x 352 y 272

Tracker 4 ☐ enable Use to calculate: ☐ T ☐ R ☐ S

track4 x 288 y 272

4. Drag the anchor over the feature in the image you want to track.

   When you move the anchor in the Viewer, the **track1, x** and **y** values change to reflect the center of the anchor.
5. In the properties panel, under **Tracker Controls**, press the track forward button to generate the tracking data from the current frame forward.
6. Inside the Tracker's properties panel, click the **Transform** tab.
7. From the **transform** list, choose an operation:

   To match another element to the current image, choose **match-move**.

   To eliminate movement (i.e., camera shake) from the current image, choose **stabilize**.

To track position, rotation, and scaling (multiple features):

1. Select the node that outputs the image you want to track.
2. Choose **Transform > Tracker** to connect a new Tracker node.
3. Click the **Tracker** tab and check the **enable** box for each of the tracks you want to activate—one for each feature you want to track in the image.

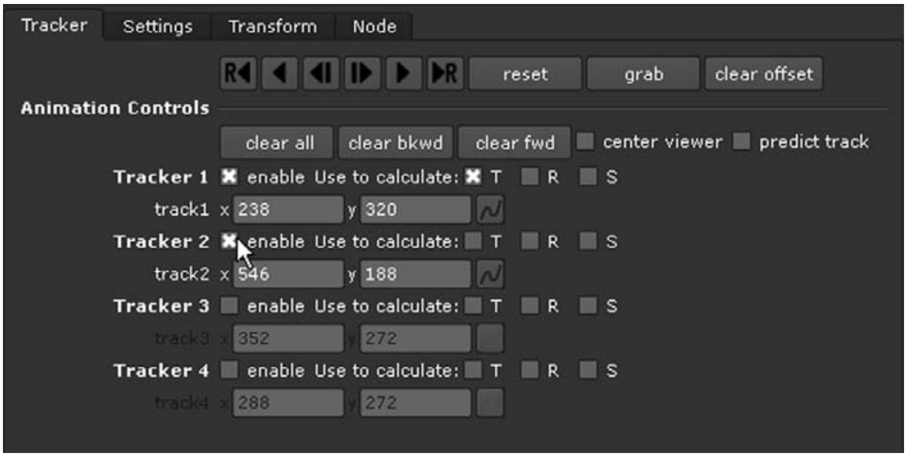
   For example, suppose you want to track the four corners of a billboard so you can matchmove a new image to it—a cornerpin track. you'll need four tracks activated.
4. Drag the anchors over the features in the image you want to track.
5. Press the track forward button in the Tracker properties panel to generate the animation data for all enabled tracks.
6. Inside the Tracker's properties panel, click the **Transform** tab.
7. Choose an operation from the **transform** list: **match-move** or **stabilize**.

Activating Track Anchors

You can select up to four track anchors. The number you choose depends on which transformational components you wish to track and the degree of accuracy you require.

To activate the tracks:

1. Click the **Tracker** tab in the properties panel.
2. Check each of the boxes for the tracks you want to **enable**.



☐ center viewer ☐ predict track

Tracker 1 ☒ enable Use to calculate: ☒ T ☐ R ☐ S
track1 x 238 y 320
Tracker 2 ☒ enable Use to calculate: ☐ T ☐ R ☐ S
track2 x 546 y 188
Tracker 3 ☐ enable Use to calculate: ☐ T ☐ R ☐ S
track3 x 352 y 272
Tracker 4 ☐ enable Use to calculate: ☐ T ☐ R ☐ S
track4 x 288 y 272

Note *After you calculate a track, you can uncheck its enable box to lock it. This protects the tracked points from being recalculated or repositioned.*

Positioning Track Anchors

A *pattern* and *search area* accompany each track anchor. The pattern area encompasses the grid of pixels that the system attempts to follow across multiple frames. This pattern should be as distinct as possible from the surrounding frame, and remain visible throughout the majority of the sequence. For example, you might choose as a pattern a high-contrast window corner which stays in frame throughout an entire shot.

The search area defines the portion of the frame in which the system looks for the pattern.

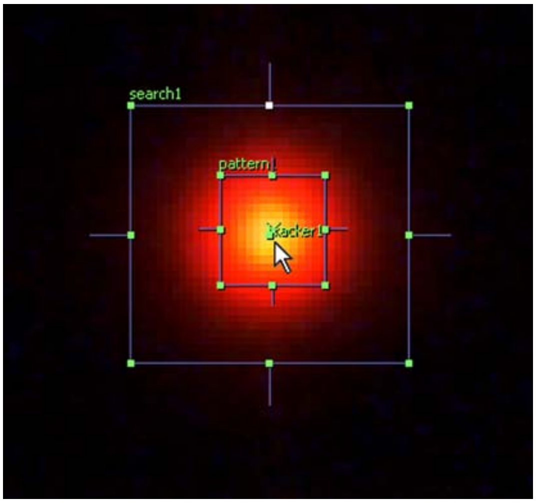


Figure 6.1: The search area contains the space where the tracker will search for the pattern. The pattern area contains the pixels that the tracker will attempt to "lock onto" for the track.

Positioning track anchors involves moving and sizing the boundaries of both the search and pattern areas. Start by moving both boundaries over the pattern to be tracked, then fine tune the position and size of each. In the end, the search area must be larger than the pattern area.

To move both the search and pattern boundaries:

1. Drag on the frame to select both boundaries with the marquee.
2. Click on the border of either boundary, then drag both over the pattern to be tracked (stop when the pattern boundary overlay's **x** sits directly on top the feature).

To adjust the position of either the search or pattern boundaries.

1. Click to the line-portion of either boundary to select it.

2. Drag to reposition the boundary.

   Or, if you're repositioning the pattern boundary, increment or decrement the track's **x** and y fields.

To adjust the size of the search or pattern boundaries

1. Click on any point on either boundary.
2. Drag to reposition the associated side.

Calculating the Track

Once you've properly placed the track anchors and sized the search and pattern areas, you're ready to calculate the track(s). You calculate tracks by using the buttons under **Tracker controls** in the Tracker properties panel. You can track the sequence in either direction. Tracking backwards can get a better track than going forwards if the feature is larger and thus more clearly visible later in the clip than at the beginning.

To toggle the tracking overlay:

If necessary, turn on the Tracker overlay in the Viewer. Click the right mouse button and choose **Overlay**, or just press **O,** over the Viewer.

Pressing **O** three times toggles between the three overlay states: **overlay off**, **overlay on**, and **overlay on, no animation path**.

To calculate tracks:

1. In the Tracker properties panel, check the **enable** box for each track you wish to calculate.
2. In the Tracker properties panel, click either the frame forward or backward buttons to move to the previous or next frame. Move through a few frames in this manner to ensure that all enabled track anchors are "sticking" to their patterns.
3. If a particular track anchor doesn't stick, experiment with a different position.
4. Once all track anchors stick, click the Tracker's track forward or track backward buttons to analyze the whole sequence.

When calculating multiple tracks simultaneously, you may find that some tracks stick with accuracy to the pattern, while others require resetting and reanalysis. When you're happy with a given track, uncheck its **enable** box. This protects it from recalculation, and lets you experiment with better placement for the wayward tracks.

If you need to start over with a given track anchor, you can reset the size of its search and pattern boxes and wipe its existing tracking data.

To reset the size of an anchor's search and pattern boxes:

1. Check the **enable** box for only the track anchor whose size you wish to reset.
2. Click the **reset** button. The track anchor's pattern and search areas are recentered to their default sizes.

To clear a track's animation data:

1. Check the **enable** box for only the track anchor whose track you wish to remove.
2. Under Animation Controls, click the **clear all** button. The selected track is removed—that is, all it's transformational data is wiped.

   To only clear animation forward or backward of the current frame, click clear fwd or clear bkwd.

Retracking Part of a Track

A tracking pattern may become unusable when it moves out of frame, is hidden by another image feature, or because of motion blur. When this happens, you can retrack the unusable part of the track with new search and pattern areas while keeping the track data consistent. The end result is a continuous track calculated from multiple patterns.

To retrack part of a track with a new search area:

1. Check the **enable** box for only the track that requires retracking.
2. Cue the Viewer to the last frame where the existing tracking is usable.
3. Ctrl+drag (Command+drag on Mac OS X) the track anchor to reposition the search and pattern areas without affecting the position of the track point. The offset allows Nuke to continue the track with the assumption that the offset feature remains at the same relative distance to the original feature.
4. Click the Tracker's track forward (or backward button, if you are tracking backwards) to continue calculating the track using the new pattern. Because the track point has been offset from the new search area, the new track values continue smoothly from the existing ones.

   ◀ ▶

Editing Tracks

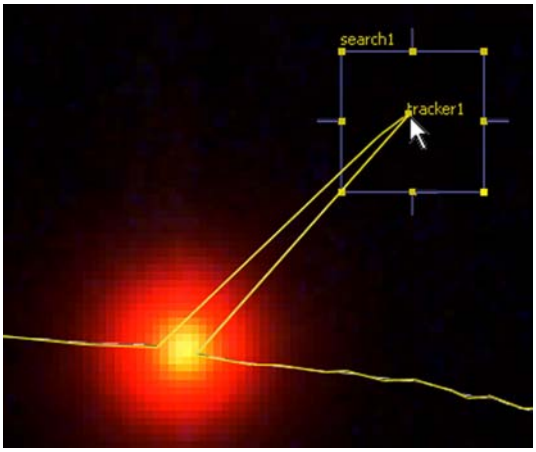
You can edit tracks via their Viewer overlays or their underlying animation curves.

Manipulating the Track Overlays

The Tracker plots existing tracks as Viewer overlays. These overlays offer an intuitive means of editing a track. If for example, you have a track in which the Tracker loses sight of its pattern for one or two frames, you can use the overlay to manually reposition the wayward track points. (This is often a faster approach than going into the associated animation curves.)

To move track points with the overlay:

1. Cue the Viewer to the frame corresponding to the track point you wish to move. The search and pattern boxes move over the point.
2. Drag the track point to the desired location. The search and pattern boxes follow.



Manipulating Track Curves and Smoothing Tracks

A track is essentially just an animated transformation matrix. Thus each track has animation curves which you can edit in order to refine a track. You can also smooth tracks using the Tracker controls.

Moving track points with curves

To move track points with curves:

1. In the Tracker properties panel, click the animation button next to the track you wish to edit, then select **Curve Editor**. The Animation editor displays the x and y curves for the track (these plot the position of each track point over time).

2. Select the points on these curves which you wish to manipulate. (Click to select individual points; drag to select multiple points with the marquee; or press **Crtl+A** to select all points.).
3. Drag the points to adjust their values. As you do so, the tracker overlay on the Viewer changes shape to reflect the new positions of the track points.

Smoothing tracks

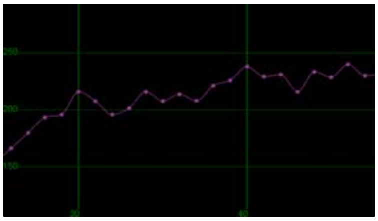
Once applied to an element, some tracks may exhibit too much jitter, which is caused by the Tracker too precisely following the pattern. You can use the Tracker controls or apply smoothing filters to a track's curves in order to remove such jitter.

To smooth tracks:

1. In the Tracker controls, go to the **Transform** tab.
2. In the **smooth** fields, enter the number of frames you want to average together to smooth the transformation. You can smooth the translate (**T**), rotate (**R**), and scale (**S**) separately.

OR

1. In the Tracker properties panel, click the **Animation menu** button next to the track you wish to edit, then select **Curve Editor**. The Animation editor displays the x and y curves for the track (these plot the position of each track point over time).
2. Select the points on these curves which require smoothing. (Click to select individual points; drag to select multiple points with the marquee; or press **Crtl+A** to select all points.).
3. Right-click on the editor and select **Edit > Filter** to apply the smoothing filter. This sets new values on each point based on the average values of their neighbouring points.
4. Enter the number of times to apply the smoothing filter in the dialog that appears. Click **OK**.
5. Reapply the smoothing filter as many times as is necessary.



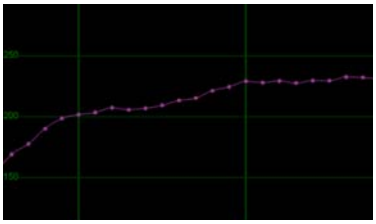


Figure 6.2: A track curve before smoothing.

Figure 6.3: A track curve after smoothing.

Tracking and Multiview Projects

If you need to use tracking data in a multiview or stereoscopic project, you may want to apply your edits to one view only (for example, the left view but not the right), or create a track in one view and have it automatically generated for the other, in the correct position.

Splitting views off

Splitting a view off allows you to edit the tracking data in that view only, without affecting any other views that exist in your project settings.

To split a view off:

1. Display the view you want to split off in the Viewer.
2. In the Tracker controls, display the **Tracker** tab. Usually, it is better to split off controls on this tab rather than the **Transform** tab. The controls on the Transform tab will compute differently per view as long as the tracks on the **Tracker** tab are different per view.
3. Click the Views menu button next to the track you want to edit and select **Split off [view name]**. For example, to edit tracking data in a view called left, select **Split off left**. Any changes you now make to the track in question are only applied to the view you chose to split off and are displaying in the Viewer.

Correlating one view from another

You can use a Tracker to track something in one view, and have the track's x and y position automatically generated for the other view.

To correlate one view from the other:

1. Track a feature in one view.
2. In the Viewer, display the view you want to generate the corresponding track for.
3. In the Tracker controls, click the Views menu button next to the track, and select **Correlate [view name] from [view name] using disparity**. For example, if you created a track for the left view and want to have the corresponding track generated for the right view, select **Correlate right from left using disparity**. This generates the corresponding track for the view you are displaying.

Tip *If you have got The Foundry's Ocula plug-ins installed, you can also do the correlation using Ocula (select* ***Correlate [view name] from [view name] with Ocula****). This way, extra refinements are done when generating the track, and the results may be more accurate.*

For more information on working with multiview projects, see Chapter 16: "Working with Stereoscopic Projects" on page 494.

Applying Tracking Data

You apply tracking data to the input image or other Nuke nodes using either the Tracker node's controls or linking expressions.

Applying Tracking Data Using Tracker Controls

The simplest way to apply tracking data to the input image or other nodes is to use the controls of the Tracker node itself. Here, we look at using these controls to stabilize or matchmove footage. If you need to apply a cornerpin track to another node, you need to do it via linking expressions.

Stabilizing elements

The Tracker node's controls let you remove motion, such as unwanted camera shake, from the node's input clip.

To stabilize the input footage:

1. Create the track you want to use for stabilizing the footage. A single track is usually enough to stabilize a feature's horizontal and vertical motion across the 2D plane. Two tracks can be used to do the same but also remove rotation in the image.
2. In the Tracker properties panel, go to the **Settings** tab. From the warp type pulldown menu, select the transformations that you want Nuke to take into account when stabilizing the image, for example **Translate/Rotate/Scale**.
3. Go to the **Transform** tab. Under **transform**, select **stabilize**.

Nuke stabilizes the footage, locking its elements to the same position within the composite.

Matchmoving elements

You can use the Tracker node's controls to apply the tracked motion to another image, that is, to matchmove an image.

To matchmove footage:

1. Use a Tracker node to create the track you want to apply to an image.
2. Copy the Tracker node and paste it after the footage you want to matchmove.
3. In the second Tracker node's controls, go to the **Transform** tab.
4. From the transform pulldown menu, choose **match-move**.

Nuke applies the tracked movement to the footage you want to matchmove.

Applying Tracking Data via Linking Expressions

Nuke's CornerPin2D and Stabilize2D nodes are specifically designed to receive tracking data via linking expressions, but you can apply tracking data in this manner to virtually any Nuke node. For example, you might animate a Bezier or a B-spline shape with tracking data by entering linking expressions into the RotoPaint node's transformation parameters. You can also apply tracking data to individual points.

This section explains the basic procedure for applying tracking data to any node via linking expressions, then discusses how to apply such data to the CornerPin2D and Stabilize2D nodes in particular.

Creating linking expressions

The Tracker node's Tracker panel displays data related to the position of each track anchor over time (**tracks' x** and **y** fields). These are the data which you most typically apply to other nodes.

To drag and drop tracking data:

1. Display both the tracker parameters (the source parameters, in this case) and the parameters to which you wish to apply the tracking data (the destination parameters—for example, a RotoPaint node's **translate** parameter).
2. **Ctrl**+drag (**Cmd**+drag on a Mac) from the source parameters animation button to the destination parameters animation button.

When you release, the destination parameters will turn blue, indicating an expression has been applied. In this case, the drag and drop action has created a linking expression resembling the following example:

```
Tracker1.tracker1.x
```

Tip *You can also apply tracking (or other transform) data to individual RotoPaint, SplineWarp, or GridWarp points (this is sometimes called per vertex tracking). To do so,* ***Ctrl/Cmd****+drag and drop the track's animation button on a RotoPaint, SplineWarp or GridWarp point in the Viewer.*

You can add other components to this linking expression as necessary. For example, you might add a spatial offset to the linking expression by subtracting out the initial frame's tracking values, in which case the final expression would resemble the following:

```
Tracker1.tracker1.x-Tracker1.tracker1.x(1)
```

See Chapter 18: "Expressions" on page 533 for more information. Once you enter the linking expression, the destination parameter turns blue.

To link animated parameters with a Tracker node

You can also link controls with the Tracker node if you use the **Link to** option in the Animation menu. For example to link the **translate** control of the Roto node with a Tracker node, do the following:

1. Create the Tracker node you want to link to.
2. Click on the translate animation menu in the Roto control panel.
3. Select **Link to** > **Tracker linking dialog**
4. Select the Tracker node you want to use in the tracker node dropdown and in the **link to** dropdown, select whether you want to link to the position of the track or the translate values of it.
5. Select which tracks you want to use by checking the **track** boxes. The Expression field updates with the appropriate expression syntax. Then click **OK**, and your linking is done.

   Your Bezier shape's translate value now changes when the Tracker value is changed.

Using the CornerPin node

The CornerPin2D node is designed to map the four corners of an image sequence to positions derived from tracking data. In practice, this node lets you replace any four-cornered feature with another image sequence. For example, suppose you needed to replace the monitor image in the fast-panning shot shown below.

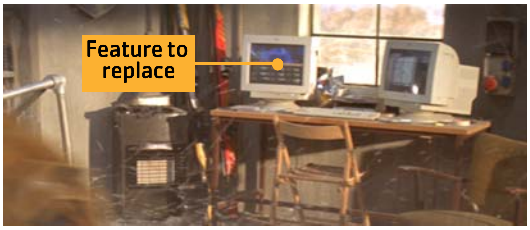


Figure 6.4: Fast-panning shot requires four corner tracking.

You would first use the Tracker to calculate four separate tracks, one for each corner of the feature.

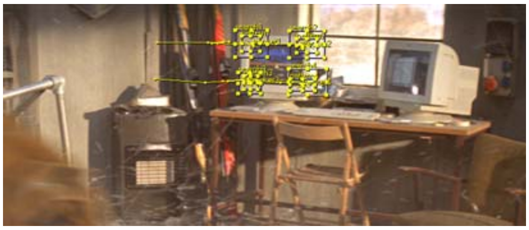


Figure 6.5: Generating the four tracks.

Next, you would attach a CornerPin2D node to the image sequence you want to use as the replacement for the feature, and apply to it the tracking data. This would remap the image sequence's corners to the correct positions over time.

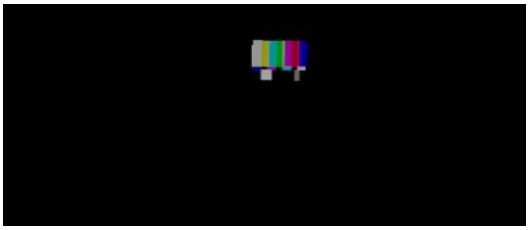


Figure 6.6: Applying the tracked corner data to the replacement image.

The final step would be to layer this result over the original element.

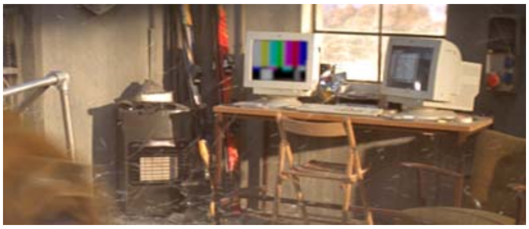


Figure 6.7: The composited image.

The steps below summarize the use of the CornerPin2D node.

To use the CornerPin2D node:

1. Use the Tracker node to generate four tracks, one per corner, on the feature requiring replacement.
2. Click **Transform > CornerPin** to add a CornerPin2D node to the script.
3. Attach the CornerPin2D node to the image sequence that will replace the feature tracked above.

   Note that the CornerPin2D node should NOT be connected to the Tracker node or the Tracker node's input image.
4. In the CornerPin2D properties panel, add linking expressions (see "Creating linking expressions" on page 256) to the positional data for the four tracks generated above.

   When linking a particular track to a particular corner, keep in mind that **to1** refers to the bottom left corner of the image sequence; **to2**, to the bottom right corner; **to3**, to the top right corner, and **to4**, to the top left corner.
5. If you want to, you can use the inverse values of the points specified in step 4 by checking the **invert** box.
6. If necessary, choose a different filtering algorithm from the **filter** pulldown menu. (See "Choosing a Filtering Algorithm" on page 227).
7. When filtering with Key, Simon, or Rifmen filters, you may see a haloing effect caused by pixel sharpening these filters employ. If necessary, check **clamp** to correct this problem.
8. In most cases, you will keep **black outside** checked. This renders as black pixels outside the image boundary, making it easier to layer the element over another. (If you uncheck this parameter, the outside area is filled with the outermost pixels of the image sequence.)

Using the Stabilize2D node

The Stabilize2D node is designed to remove unwanted camera movement, rotation, and/or scaling from an image sequence. The node requires data from only a single track if you only need to stabilize movement; it requires data from two tracks if you need to stabilize for rotation and/or scaling.

The basic procedure for using Stabilize2D is to first use the Tracker node to generate the required tracks, then follow the Tracker node with a Stabilize2D node. To this node, you apply the tracking data in inverse form, thus negating the unwanted transformations.

To use the Stabilize2D node:

1. Use the Tracker node to generate the appropriate number of tracks on the element requiring stabilisation. Remember, you'll need at least two tracks if you need to stabilize for more than just movement. (You can, of course, generate more tracks and average the results for better accuracy.)
2. Select the Tracker node used above, then click **Transform > Stabilize** to add a Stabilize2D node to the Tracker node.
3. From the **type** pulldown menu in the Stabilize2D properties panel, select:
    - **1 Point** to stabilize only for movement.
    - **2 Point** to stabilize for rotation and/or scaling.
4. Check all transformation types which you wish to cancel out.
5. In the both the **track1 x** and **y** fields, type **1-** (to invert the data), followed by a linking expression to the relevant tracking data in the Tracker node used above. (You can use either the positional data for Tracker 1, or some multiple track average from the Outputs panel.)

   Your entries should resemble the following examples: **1-Tracker1.tracker1.x** and **1-Tracker1.tracker1.y**.
6. Repeat the above for the **track2 x** and **y** fields.
7. As you apply the tracking data, the current frame displayed in the Viewer is likely to move out of view. This is because the node applies the inverted tracking data to the bottom left corner of the image sequence. Enter values in the **offset XY x** and **y** fields to restore the image to the center of frame. (You may have to animate these values over time to keep the image centered.)
8. If necessary, choose a different filtering algorithm from the **filter** pulldown. (See "Choosing a Filtering Algorithm" on page 227).
9. When filtering with Key, Simon, or Rifmen filters, you may see a haloing effect caused by pixel sharpening these filters employ. If necessary, check **clamp** to correct this problem.
10. In most cases, you will keep **black outside** checked. This renders as black pixels outside the image boundary. (If you uncheck this parameter, the outside area is filled with the outermost pixels of the image area.)

7 KEYING WITH PRIMATTE

This section explains how to use the blue/green screen keyer, Primatte, in Nuke.

Accessing Primatte from Nuke

Start up Nuke, create a Primatte node and connect a foreground and a background image to it. Add a Nuke Viewer node so you can see the result.

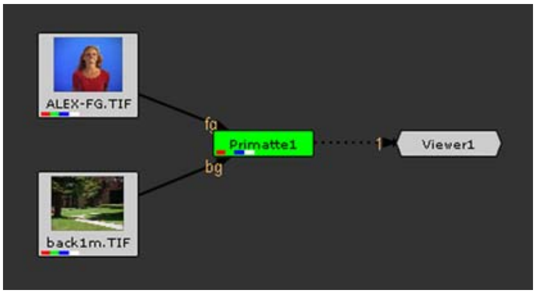
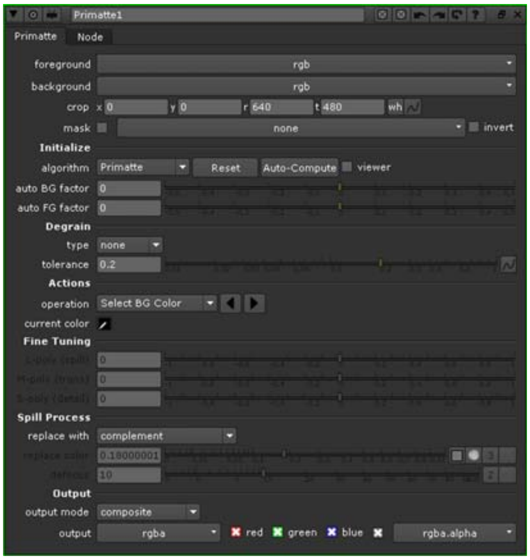

ALEX-FG.TIF
back1m.TIF
fg
bg
Primatte1
1
Viewer1

Figure 7.1: Nuke flowgraph.

When you select the Primatte node, you should be presented with the Primatte properties panel as shown below.


Primatte1
Primatte
Node
Initialize
Reset
Auto-Compute
Degrain
Actions
Fine Tuning
Spill Process
Output

foreground rgb
background rgb
crop x 0 y 0 r 640 t 480 wh
mask ☐ none ☐ invert
algorithm Primatte ☐ viewer
auto BG factor 0
auto FG factor 0
type none
tolerance 0.2
operation Select BG Color
L-poly (spill) 0
M-poly (trans) 0
S-poly (detail) 0
replace with complement
replace color 0.18000001
defocus 10
output mode composite
output rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha

Figure 7.2: Primatte properties panel.

Primatte Basic Operation Tutorial

This describes the operation of the Primatte node in Nuke. A more detailed explanation of how the Primatte algorithm actually works can be found under "The Primatte Algorithm" on page 287.

Auto-Compute

This version of Primatte has a new feature that may eliminate the first three steps of using earlier versions of Primatte. It is called the **Auto-Compute** button and may make your keying operation much easier. You can click on this button as a first step and it may automatically sense the backing screen color, eliminate it and even get rid of some of the foreground and background noise that would normally be cleaned up in Step 2 (**Clean BG Noise**) and Step 3 (**Clean FG Noise**) of the Primatte operation. If you get good results then jump ahead to the spill removal tools.

The **Auto-Compute** button has two sliders that modify its behavior; the **auto BG factor** and the **auto FG factor** sliders. These may be moved to get better results with the **Auto-Compute** button. This is useful when doing a set of clips that have similar backgrounds and lighting. Once the sliders are configured for a particular lighting set-up, all the clips will key quickly using just the **Auto-Compute** button.

If you don't get the results you wanted from **Auto-Compute**, please continue from this point on to get the basic Primatte operation procedures.

The basic functionality for the Primatte interface is centered around the **Actions** or **operation** pulldown menu and the Viewer window.

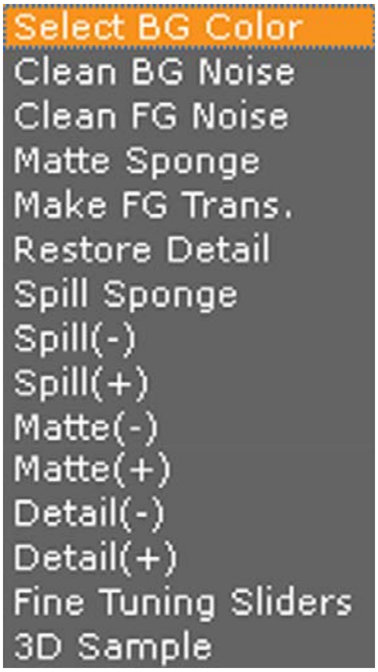

Select BG Color
Clean BG Noise
Clean FG Noise
Matte Sponge
Make FG Trans.
Restore Detail
Spill Sponge
Spill(-)
Spill(+)
Matte(-)
Matte(+)
Detail(-)
Detail(+)
Fine Tuning Sliders
3D Sample

Figure 7.3: Primatte operation menu.

There are four main steps to using the Primatte and **Select BG Color** is the first step.

Select BG Color

Ensure that the **Select BG Color** action is selected (it should be at this time as it is the default **Action** mode when you start Primatte).

Position the cursor in the bluescreen area (or whatever background color you are using), usually somewhere near the foreground object. Hold the **Ctrl/Cmd** key down and sample the targeted background color. Release the mouse button and Primatte will start the compositing process. If the foreground shot was done under ideal shooting conditions, Primatte will have done 90-95% of the keying in this one step and your image might look like this.

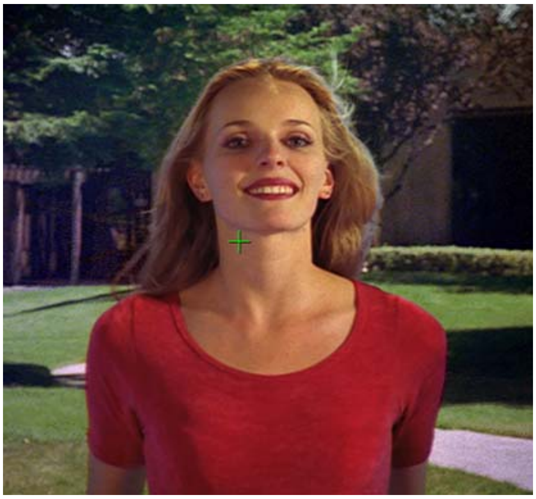


Figure 7.4: Basic key.

Note *Primatte will work equally well with any color backing screen. It does not have to be a specific shade of green or blue.*

Tip *If you dragged the cursor in the blue area, Primatte averages the multi-pixel sample to get a single color to adjust to. Sometimes Primatte works best when only a single pixel is sampled instead of a range of pixels. The color selected at this point in the Primatte operation is critical to the operation of the node from this point forward. Should you have difficulties further along in the tutorial after selecting a range of blue shades, try the Select BG Color operation again with a single dark blue pixel or single light blue pixel. You can also switch to the alpha channel view and click around in the bluescreen area and see the different results you get when the initial sample is made in different areas.*

Tip *If you would rather make a rectangular selection and not use the default 'snail trail' sampling method, you can do a **Ctrl+Shift+**drag sample.*

Tip *If the foreground image has a shadow in it that you want to keep it in the composite, do not select any of the dark blue pixels in the shadow and the shadow will come along with the rest of the foreground image.*

The second and third steps in using Primatte require viewing the **Matte or Alpha** view in the **Viewer** window. Press the 'A' key on the keyboard to change to the **Alpha** view. The image displayed will change to a black and white 'matte' view of the image that looks like this.

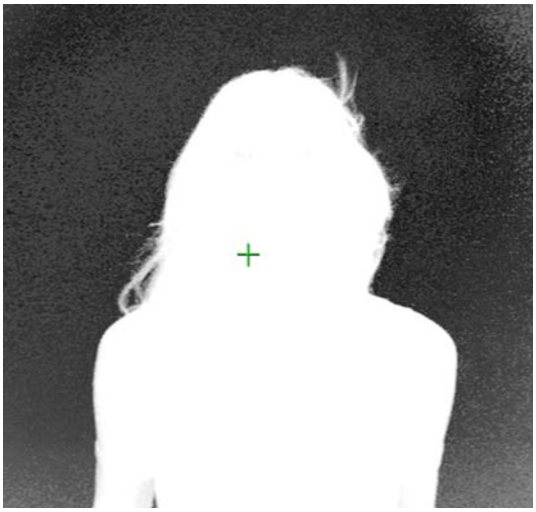


Figure 7.5: Matte.

Clean BG Noise

Change the **Actions Operation** from **Select BG Color** to **Clean BG Noise**. If there are any white regions in the dark, `bluescreen area', it is 'noise' (or shades of blue that did not get picked up on the first sample) and should be removed. Sample through these whitish noise regions and when you let up on the pen or mouse button, Primatte will process the data and eliminate the noise. Repeat this procedure as often as necessary to clear all the noise from the background areas. Sometimes increasing the brightness of your monitor or the screen gamma allows you to see noise that would otherwise be invisible.

Note *You do not need to remove every single white pixel to get good results. Most pixels displayed as a dark color close to black in a key image will become transparent and virtually allow the background to be the final output in that area. Consequently, there is no need to eliminate all noise in the bluescreen portions of the image. In particular, if an attempt is made to meticulously remove noise around the foreground object, a smooth composite image is often difficult to generate.*

Tip *When clearing noise from around loose, flying hair or any background/ foreground transitional area, be careful not to select any of areas near the edge of the hair. Leave a little noise around the hair as this can be cleaned up later using the Fine Tuning Sliders tool.*

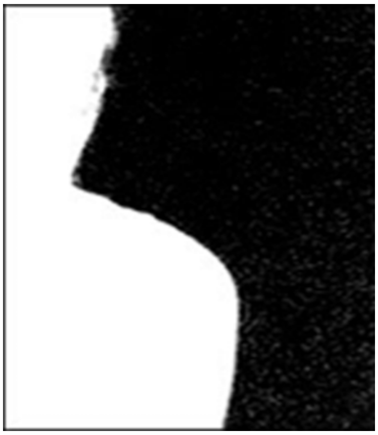


Figure 7.6: Before background noise removal.

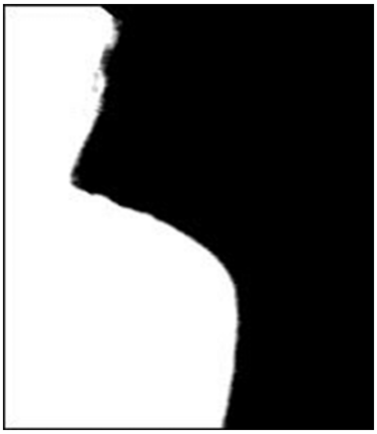


Figure 7.7: After background noise removal.

Clean FG Noise

If there are dark regions in the middle of the mostly white foreground object, that is, if the key is not 100% in some portion of the targeted foreground, choose **Clean FG Noise** from the **Actions operation** pop-up menu. Use the same techniques as for **Clean BG Noise**, but this time sample the dark pixels in the foreground area until that area is as white as possible.

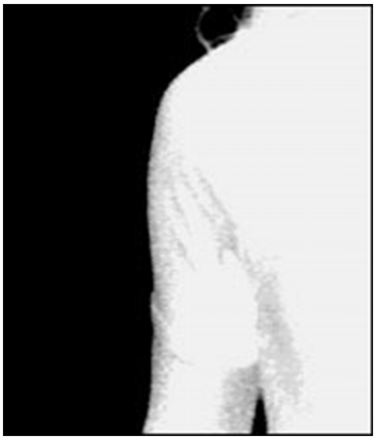


Figure 7.8: Before foreground noise removal.

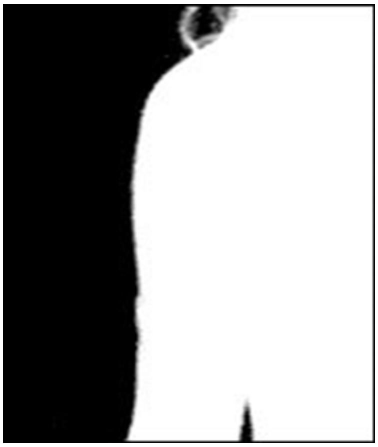


Figure 7.9: After foreground noise removal.

These were the steps necessary to create a clean 'matte' or 'key' view of the image. With this key, the foreground can be composited onto any background image. However, if there is `spill' on the foreground object from light that was reflected off the background, a final operation is necessary to remove that background spill get a more natural looking composite.

For the fourth step in the Primatte operation, return the **RGB** view to the monitor window by clicking again on the **`A'** keyboard key. This will turn off the alpha channel viewing mode and the Viewer window will again display the **RGB** view with the background image (if you connected one to the Primatte node).

The sample image below has gone through the first three steps and has examples of spill. Notice the blue fringe to her hair and a blue tint on her right cheek, arm and chest.

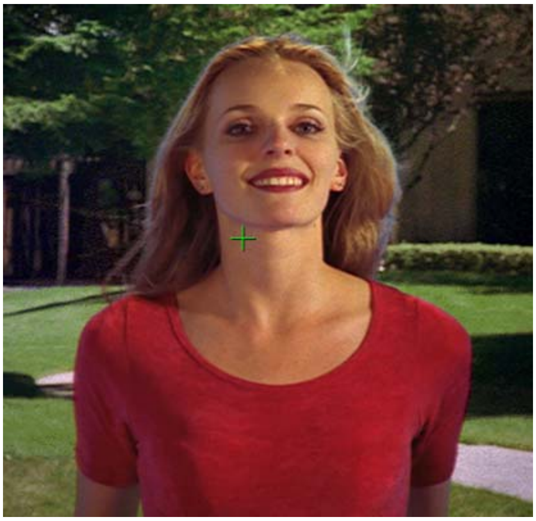


Figure 7.10: Blue spill visible.

Spill Removal - Method #1

There are three ways in Primatte to remove the spill color. The quickest method is to select the **Spill Sponge** button from the **Actions operation** area and then sample the spill areas away. By just positioning the cursor over a bluish pixel and sampling it, the blue will disappear from the selected color region and be replaced by a more natural color. Additional spill removal should be done using the **Fine Tuning** tools or by using the **Spill(-)** feature. Both are explained further on in this manual.

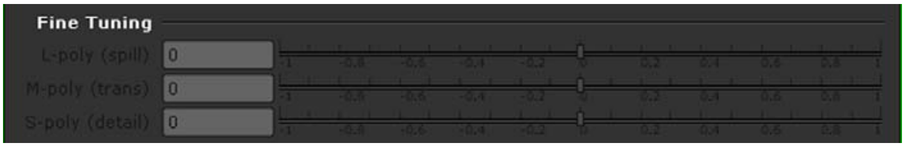
Note *All spill removal/replacement operations in Primatte can be modified using the **Spill Process 'replacement with'** tools. Spill can be replaced with either* *the **complement** of the background color, a **solid color** selected by the user or by colors brought from a **defocused background.** Depending on the spill conditions, one of these options should provide the results you are looking for. See the information in "Spill Replacement Options" on page 270 for more details.*

Note *Primatte's spill removal tools work on 'color regions'. In the image above, samples should be made on the light flesh tones, the dark flesh tones, the light blonde hair, the dark blonde hair and the red blouse color regions. One sample in each color region will remove spill from all similar colors in the foreground image.*

If the spilled color was not been totally removed using the **Spill Sponge** or the result of the **Spill Sponge** resulted in artifacts or false coloring, a fine-tuning operation **Spill(-)** tool should be used instead for a more subtle and sophisticated removal of the spilled background color. This is discussed in "Spill (-)" on page 283.

Spill Removal – Method #2

1. Select the **Fine Tuning Sliders Actions operation**. This will activate these tools.



Fine Tuning

L-poly (spill) 0

M-poly (trans) 0

S-poly (detail) 0

2. Using the zoom and pan capabilities of the Nuke application, zoom into an area that has some blue edges or spill.
3. Using the cursor, sample a color region that has some spill in it. When you let up on the pen or mouse button, Primatte will register the color selected (or an average of multiple pixels) in the **current color** area. For most images, the **L-poly (spill)** slider is all that is required to remove any remaining blue spill. The more to the right the slider moves, the more spill color will be removed from the sampled pixels. The more to the left the slider moves, the more the selected pixels will move toward the color in the original foreground image.

   Note: When using the **L-poly (spill)** slider, spill color replacement will be replaced based on the setting of the **Spill Process 'replacement with'** settings. For more information on these tools, see the section of this manual on **Spill Replacement** in **Chapter 6. Primatte Tools and Buttons**.

Tip *It is better to make several small adjustments to the blue spill areas than a single major one.*

4. You can use the other two sliders in the same way for different key adjustments. The **S-poly (detail)** slider controls the matte softness for the color which is closest to the background color. For example, you can recover lost rarefied smoke in the foreground by selecting the **Fine Tuning Sliders** action, sampling the area of the image where the smoke just starts to disappear and moving the **S-poly (detail)** slider to the left. The **M-poly (trans)** slider controls the matte softness for the color which is closest to the foreground color. For example, if you have thick and opaque smoke in the foreground, you can make it semi-transparent by moving the **Transparency** slider to the right after selecting the pixels in the **Fine Tuning Sliders** mode.

Tip *If the foreground image changed color dramatically during the fine tuning process, you can recover the original color by selecting an area of the off-color foreground image and moving the L-poly (spill) slider slightly to the left. This may introduce spill back into that color region. Again, use the Fine Tuning Sliders option to suppress the spill, but make smaller adjustments this time.*

Spill Removal – Method #3

1. This method uses a more recent Primatte tool that is covered in detail in *Repeatable Sampling Tools* below.

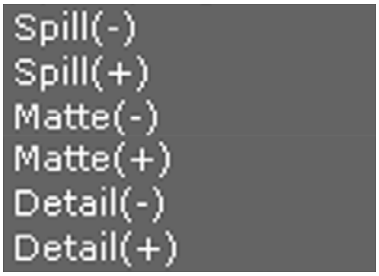
Note *If these final 'spill suppression' operations have changed the final compositing results, you may have to return to earlier operations to clean up the matte. If the composite view looks good, it is a good idea to go back and take a final look at the alpha channel view. Sometimes in the Primatte operation, a 100% foreground area (all white) will become slightly transparent (gray). You can clean those transparent areas up by using the Matte Sponge tool. After selecting the Matte Sponge tool, just click on the transparent pixels and they will become 100% foreground. All of the spill-suppression information will remain intact. Alternatively, you can go to the alpha channel view and then using the Fine Tuning Sliders option, select those transparent areas and move the Transparency slider slightly to the left. This will move that color region from 0-99% foreground with spill suppression to 100% foreground with spill suppression and should solve the problem. The Matte(+) tool will also work to solve this problem.*

Repeatable Sampling Tools

Most of the Primatte operations are done using a 'mouse or pen sampling' operation. The only exceptions are the **Fine Tuning Sliders** Actions mode and its sliders. The **Fine Tuning Sliders** action mode gives a continuous valuator for fine-tuning but some of the sliders are not often used because results are often unpredictable or not subtle enough.

Another weak point in previous versions of Primatte is the lack of functionality to attenuate and thicken the existing matte density. This version of Primatte offers a more intuitive, easy-to-use and powerful user interface called **Repeatable Sampling**.

In addition to the conventional Primatte operation modes previously mentioned, six other tools are added:



Spill(-)
Spill(+)
Matte(-)
Matte(+)
Detail(-)
Detail(+)

The Spill Sampling Tools

Using the **Spill(+)** and **Spill(-)** modes, you can gradually remove or recover the spill intensity on the foreground object by sampling the referenced color region repeatedly. The conventional **Spill Sponge** tool removes the spill component in a single action at one level and did not allow sampling the same pixel a second time. Even though just a small amount of spill needed to be removed, the spill sponge removed a preset amount without allowing any finer adjustment.

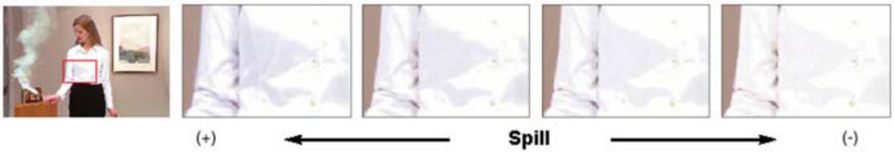


Figure 7.11: The effect of Spill (+/-) repeatable sampling.

Using the zoom and pan capabilities of the Nuke application, zoom into an area that has some blue edges and click on a pixel with some spill on it. Repeated clicking will incrementally remove the spill. Continue this operation until the desired result is achieved.

The Matte Sampling Tools

The **Matte(+)** and **Matte(-)** modes are used to thicken or attenuate the matte information. If you want a thinner shadow on a foreground object, you can use the **Matte(-)** mode as many times as you like to make it more transparent. On the other hand, you can use the **Matte(+)** mode to make the matte thicker in that color region.

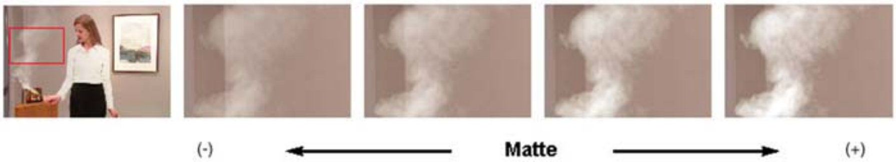


Figure 7.12: Effect of Matte (+/-) Repeatable Sampling.

The Detail Sampling Tools

The **Detail(+)** and **Detail(-)** modes are a refined version of **Clean BG Noise** and **Restore Detail** (discussed later in this document). For example, when you see some dilute noise in the backing area but don't want to remove it completely because it affects some fine detail in a different area, try using **Detail(-)**. It will attenuate the noise gradually as multiple samples are made on the pixel. You should stop the sampling when important fine details start to disappear.

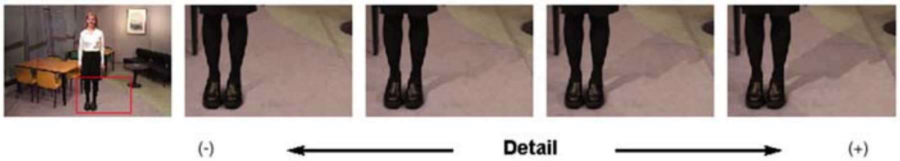
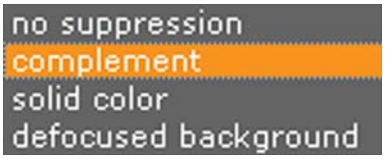


Figure 7.13: Effect of Detail (+/-) Repeatable Sampling.

Spill Replacement Options

The proper processing of spill on foreground objects is one of the many useful features of Primatte. You can move between these four modes to see how they affect the image clip you are working with. The four methods are as follows:



1. No Suppression (no suppression)
2. Complemental Spill Replacement (complement)
3. Solid Color Spill Replacement (solid color)
4. Defocus Spill Replacement (defocused background)

No Suppression (No Suppression)

In this mode, no suppression is applied.

Complemental Replacement Mode (Complement)

This is the default spill replacement mode. This mode will maintain fine foreground detail and deliver the best quality results. If foreground spill is not a major problem, this mode is the one that should be used.

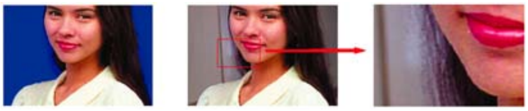


Figure 7.14: Complemental Replacement mode maintains fine detail.

The **Complemental Replacement** mode is sensitive to foreground spill. If the spill intensity on the foreground image is rather significant, this mode may often introduce serious noise in the resultant composite.

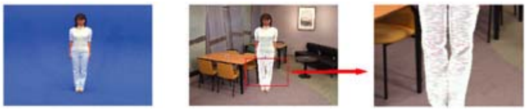


Figure 7.15: Serious noise in the composite.

Solid Color Replacement Mode (Solid Color)

In the **Solid Color Replacement** mode, the spill component will be replaced by a 'user defined' palette color. While the **Complemental Replacement** mode uses only the backing color complement to remove small amounts of spill in the original foreground, the **Solid Color Replacement** mode tries to assuage the noise using the 'user defined' palette color. Changing the palette color for the solid replacement, the user can apply good spill replacement that matches the composite background. Its strength is that it works fine with even serious blue spill conditions.

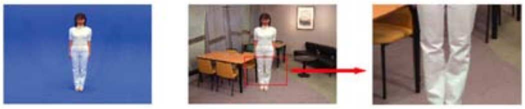


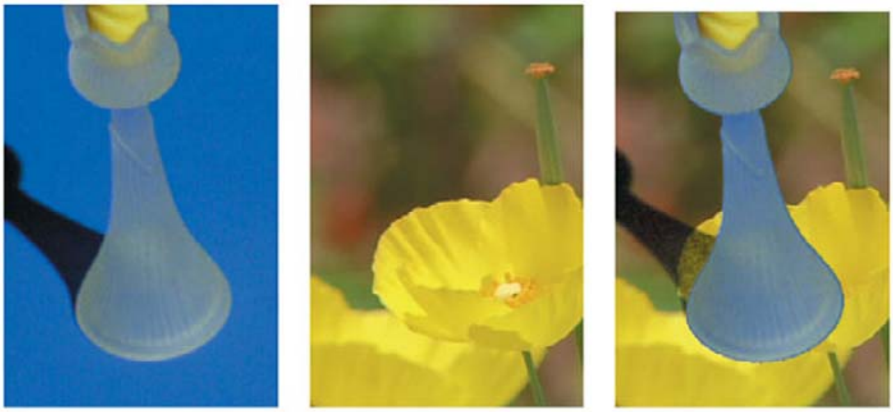
Figure 7.16: Smooth spill processing with solid color replacement.

On the negative side, when using the **Solid Color Replacement** mode, fine detail on the foreground edge tends to be lost. The single palette color sometimes cannot make a good color tone if the background image has some high contrast color areas.

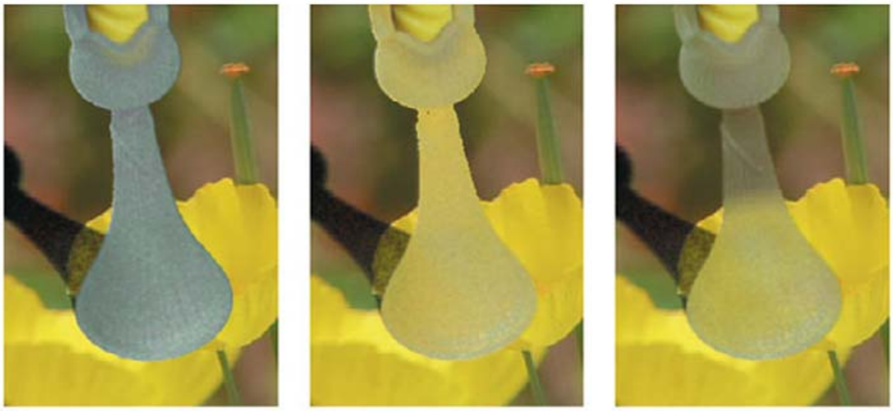
Defocus Spill Replacement (Defocused Background)

The **Defocus Replacement** mode uses a defocused copy of the background image to determine the spill replacement colors instead of a solid palette color or just the complement color. This mode can result in good color tone on the foreground object even with a high contrast background. As in the example below, spill can even be removed from frosted glass using this feature and still retain the translucency.

On the negative side, the **Defocus Replacement** mode sometimes results in the fine edge detail of the foreground objects getting lost. Another problem could occur if the user wanted to later change the size of the foreground image against the background. Since the background/foreground alignment would change, the applied color tone from the defocused image might not match the new alignment.



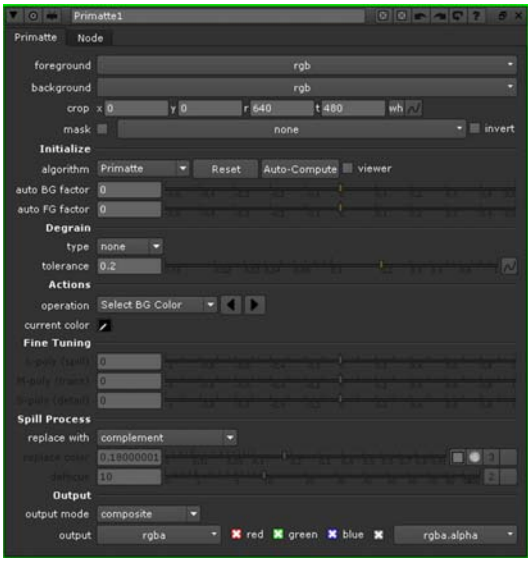
(a)Original FG Object (b)Original BG (c)Spill Image



(d)Complemental Model (e)Yellow replacement (f)Defocus Replacement

Figure 7.17: Blue suppression of a frosted glass object.

Primatte Tools and Buttons



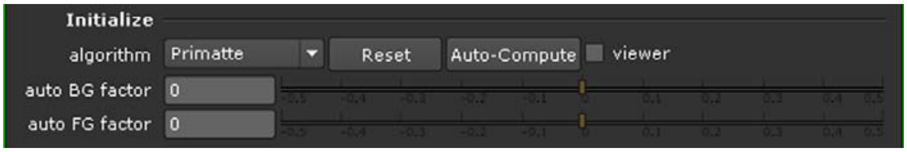
Primatte1
Primatte
Node
Initialize
Reset
Auto-Compute
viewer
Degrain
Actions
Fine Tuning
S-poly (spill)
M-poly (trans)
D-poly (detail)
Spill Process
replace color
defocus
Output
invert
red
green
blue

foreground rgb
background rgb
crop x 0 y 0 r 640 t 480
mask none
algorithm Primatte
auto BG factor 0
auto FG factor 0
type none
tolerance 0.2
operation Select BG Color
current color
S-poly (spill) 0
M-poly (trans) 0
D-poly (detail) 0
replace with complement
replace color 0.18000001
defocus 10
output mode composite
output rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha

Figure 7.18: Primatte node controls.

Initialize Section

Primatte algorithm



Initialize
Reset
Auto-Compute
viewer

algorithm Primatte ☐ viewer
auto BG factor 0
auto FG factor 0

Figure 7.19: Primatte algorithm menu

The **Primatte** algorithm mode delivers the best results and supports both the **Solid Color** and the **Complement Color** spill suppression methods. It is the algorithm that uses three multi-faceted polyhedrons (as described further down in the this document) to separate the 3D RGB colorspace. It is also the default algorithm mode and, because it is computationally intensive, it may take the longest to render.

Primatte RT+ algorithm

Primatte RT+ is in between the above two options. It uses a six planar surface color separation algorithm (as described further down in the this document) and will deliver results in between the other two in both quality and performance. Other disadvantages of the **Primatte RT+** algorithm is that it does not work well with less saturated backing screen colors and it does not support the **Complement Color** spill suppression method.

Primatte RT algorithm

Primatte RT is the simplest algorithm and therefore, the fastest. It uses only a single planar surface to separate the 3D RGB colorspace (as described further down in the this document) and, as a result, does not have the ability to separate out the foreground from the backing screen as carefully as the above Primatte algorithm. Other disadvantages of the **Primatte RT** algorithm is that it does not work well with less saturated backing screen colors and it does not support the **Complement Color** spill suppression method.

Reset

Resets all of the Primatte key control data back to a blue or greenscreen.

Auto-Compute

The **Auto-Compute** button can be used as the first step in the Primatte operation. It's purpose is to try and do the first three steps of the Primatte operation for you. It will try to automatically detect the backing screen color, remove it and do some clean-up on the foreground and background noise. If the clip was shot with an evenly lit, well saturated backing screen, the **Auto- Compute** button will leave you with an image that may only need some spill removal to complete your keying operation.

auto FG Factor

The **auto FG Factor** slider can be used to modify how the Auto-Compute algorithm deals with foreground noise. Change the position of this slider and you can see the results of the **Auto-Compute** operation change.

auto BG Factor

The **auto BG Factor** slider can be used to modify how the Auto-Compute algorithm deals with background noise. Change the position of this slider and you can see the results of the **Auto-Compute** operation change.

3D Viewer

This selector opens a window in the Viewer that displays a graphical representation of the Primatte algorithms and allows the user to see what is happening as the various Primatte tools are used. It is a passive feature that has no adjustment capabilities, it may prove useful in evaluating an image as you operate on it.

When you select it, you are presented with a window that may look similar to one of these images (depending on which Primatte algorithm you have selected).

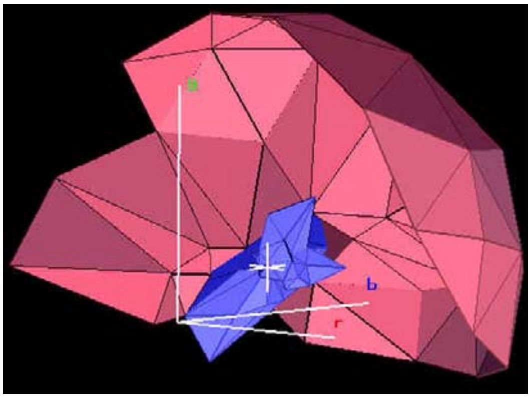


Figure 7.20: Primatte algorithm.

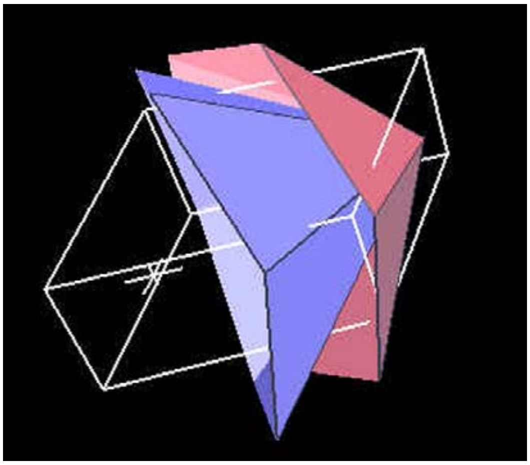


Figure 7.21: Primatte RT+ algorithm.

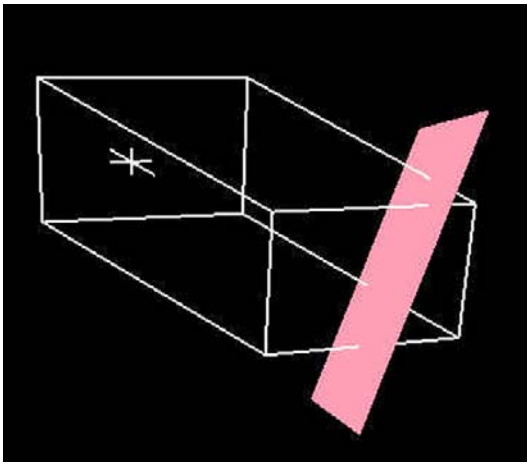


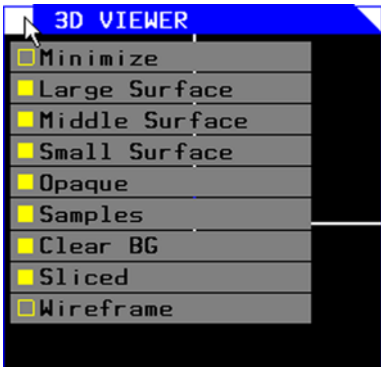
Figure 7.22: Primatte RT algorithm.

The different algorithms are described in more detail in a later section of this manual. Here is a description of the tools and features of the 3D Viewer:

3D Viewer tools

At the top of the 3D Viewer window are three areas that can be clicked on:

1. Clicking and dragging on the blue center area allows the user to move the window around on the screen.
2. Clicking and dragging on the triangular white region in the upper right corner allows the user to scale the 3D Viewer window.
3. Clicking on the square white region in the upper left of the window displays a pop-up menu that looks like this:



3D VIEWER
☐ Minimize
■ Large Surface
■ Middle Surface
■ Small Surface
■ Opaque
■ Samples
■ Clear BG
■ Sliced
☐ Wireframe

Note *A 'selected feature has a solid yellow square next to it. An 'unselected' feature has a hollow yellow square next to it.*

Minimize

This feature, when selected, makes the 3D Viewer window disappear. Only the blue title bar at the top of the window remains.

Large Surface

This feature, when selected, displays the large Primatte polyhedron in the Viewer window.

Middle Surface

This feature, when selected, displays the medium Primatte polyhedron in the Viewer window.

Small Surface

This feature, when selected, displays the small Primatte polyhedron in the Viewer window.

Opaque

This feature, when selected, makes the selected polyhedrons opaque. De-selecting it makes them semi-transparent.

Samples

This feature, when selected, allows the user to sample color regions on the image window using the 3D Sample Actions mode and see where those regions are in relation to the polyhedron and the algorithm. The colors will be displayed as a spray of pixels in the color selected. This button only allows the user to see or hide the sampled colors.

Note *The 3D Sample Actions mode must be selected in the Actions area for this feature to operate.*

Clear BG

This feature changes the background color of the 3D Viewer window from black (when unselected) to transparent (when selected).

Sliced

This feature, when selected, slices open the large and medium polyhedrons so that the inner polygons can be seen. When unselected, the largest polyhedron selected becomes a completely closed polyhedron and you might not be able see the inner polyhedrons (unless the **Opaque** feature is deselected).

Wireframe

This feature, when selected, changes the polyhedrons from shaded-surface objects to wireframe objects.

Degrain Section

Degrain tools

The **Degrain** tools are used when a foreground image is highly compromised by film grain. As a result of the grain, when backing screen noise is completely removed, the edges of the foreground object often become harsh and jagged leading to a poor key. These tools were created to, hopefully, help when a compositing artist is faced with a grainy image.

Degrain type

The **Degrain type** selector gives the user a range of grain removal from 'none' to 'large'. If the foreground image has a large amount of film grain induced pixel noise, you may lose a good edge to the foreground object when trying to clean all the grain noise with the **Clean BG Noise Actions** mode. These tools allow the user to clean up the grain noise without affecting the quality of the key. A short tutorial explaining when and how to use these tools is at the end of this section.

none

When **none** is selected, the user gets the color of the exact pixel sampled. This is the default mode.

small

When **small** is selected, the user gets the average color of a small region of the area around the sampled pixel. This should be used when the grain is very dense.

medium

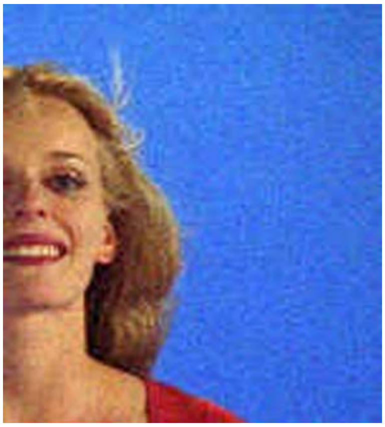
When **medium** is selected, the user gets the average color of a medium-sized region of the area around the sampled pixel. This should be used when the grain is less dense.

large
When **large** is selected, the user gets the average color of a larger region of the area around the sampled pixel. This should be used when the grain is very loose.

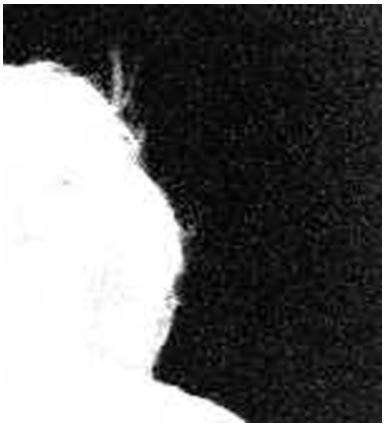
Tolerance slider
Adjusting the **tolerance** slider this should increase the effect of the **Clean BG Noise** tool without changing the edge of the foreground object.

Degrain tools tutorial

If you have a noisy image as in the example below...



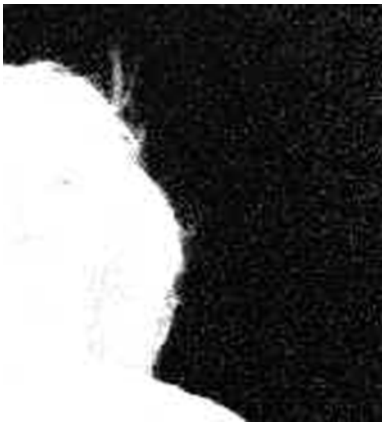
...you will find that the matte is also noisy:



Currently you can use the **Clean BG Noise** operation to remove the noisy pixels, but this can also modify the edge of the foreground object in a negative manner.

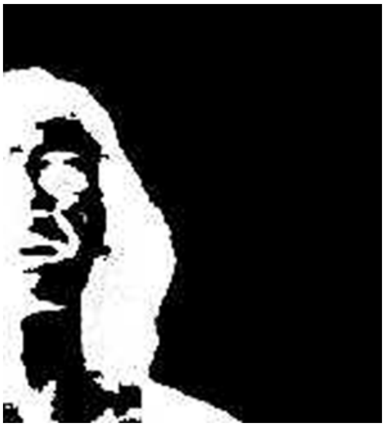
Using the **Degrain Tools** in the following way may help you clean up the image and still get a good edge on the matte:

1. Use the **Clean BG Noise** operation just a small amount to remove some of the white noise in the **Alpha** channel view but do use it so much that you affect the edge of the foreground object.
2. Then select the **Grain Size** tool and select **small** as a first step to reduce the grain:



With the degrain **tolerance** slider set at **0**, move it around some. This should increase the affect of the **Clean BG Noise** tool without changing the edge of the foreground object.

Sometimes this may not be enough to totally remove the grain so by adjusting the degrain **tolerance** slider, you can tell the Primatte algorithm what brightness of pixels you think represents grain. You should try not to use too high of a value otherwise it will affect the overall matte. For an example of an 'over adjusted' image see below.



The Primatte degrain algorithm uses a 'Defocused Foreground' image to compute the noise.

Note *The **small**, **medium** and **large** settings for the degrain tools all produce defocused foregrounds that have larger or smaller blurs respectively.*

Note *It is important to make sure that the crop settings are correctly applied otherwise when the defocus image is generated, if there is 'garbage' on the edges of the images, then that garbage will be blurred into the defocus foreground.*

As a review:

1. Select the **Select BG Color Actions** mode and click on a backing screen color.
2. Select the **Clean BG Noise Actions** mode and use it sparingly so that it has minimum affect to the edge of the foreground object.
3. If there is still grain in the backing screen area, then use the degrain **type** functionality starting at the **small** setting to reduce the grain
4. If the grain is still present, then try increasing the **tolerance** slider a little - not too much.
5. If grain is still a problem, then try changing the **type** to **medium** or **large** and also changing the grain tolerance until the desired effect is achieved.

Note *The grain functionality does not always remove grain perfectly but is sometimes useful to minimize its effects.*

Actions Section

Actions operation tools/modes

Select Background Color

When this operational mode is selected, the Primatte operation will be initially computed by having the user sample the target background color within the image window. For keying operations, this is the first step and should be followed by the steps described immediately below.

Clean Background Noise

When this operational mode is selected, the user samples pixels on the image window known to be 100% background. White noisy areas in the

100% background region will become black. This is usually the second step in using Primatte.

Clean Foreground Noise

When this operational mode is selected, the user samples pixels on the image window known to be 100% foreground. The color of the sampled pixels will be registered by Primatte to be the same color as in the original foreground image. This will make dark gray areas in the 100% foreground region become white. This is usually the third step in using Primatte.

Spill Sponge

When this operational mode is selected, the background color component in the sampled pixels (or spill) within the image window is keyed out and removed for the color region selected. This operation can only be used once on a particular color and the amount of spill suppression applied is not adjustable. It is the fastest way to remove spill from a composite image. For more accurate spill suppression, a **Fine Tuning** or **Spill (+)** operation should follow or be used instead. This can usually be the fourth (and final) step in using Primatte unless additional adjustments are necessary.

Matte Sponge

When this operational mode is selected, the sampled color within the image window becomes 100% foreground. However, if the sampled color is already keyed out and removed, it leaves the current 'suppressed' color. It only affects the key or matte information. This tool is usually used to quickly remove stray transparent pixels that have appeared during the chromakeying procedure. It is a quick and easy way to make final adjustments to a composite.

Restore Detail

With this mode selected, the completely transparent background region sampled in the image window becomes translucent. This operation is useful for restoring lost hair details, thin wisps of smoke and the like. It shrinks the small polyhedron slightly.

Make Foreground Transparent

When this mode is selected, the opaque foreground color region sampled in the image window becomes slightly translucent. This operation is useful for the subtle tuning of foreground objects which are otherwise 100 percent covered with smoke or clouds. It can only be used one time on a particular color. For a more flexible way to thin out a color region and be able to make multiple samples, you should use the **Matte (-)** tool. It expands the medium polyhedron slightly.

Spill (+)

When this operational mode is selected, color spill will be returned to the sampled pixel color (and all colors like it) in the amount of one Primatte increment. This tool can be used to move the sampled color more in the direction of the color in the original foreground image. It can be used to nullify a **Spill (-)** step. This tool dents the Primatte large polyhedron in the color region sampled.

Spill (-)

When this operational mode is selected, color spill will be removed from the sampled pixel color (and all colors like it) in the amount of one Primatte increment. If spill color remains, another click using this operational mode tool will remove more of the color spill. Continue using this tool until all color spill has been removed from the sampled color region. This tool expands the Primatte large polyhedron in the color region sampled.

Matte (+)

When this operational mode is selected, the matte will be made more opaque for the sampled pixel color (and all colors like it) in the amount of one Primatte increment. If the matte is still too translucent or thin, another click using this operational mode tool will make the sampled color region even more opaque. This can be used to thicken smoke or make a shadow darker to match shadows in the background imagery. It can only make these adjustments to the density of the color region on the original foreground image. It can be used to nullify a **Matte (-)** step. This tool dents the Primatte medium polyhedron in the color region sampled.

Matte (-)

When this operational mode is selected, the matte will be made more translucent for the sampled pixel color (and all colors like it) in the amount of one Primatte increment. If the matte is still too opaque, another click using this operational mode tool will make the sampled color region even more translucent. This can be used to thin out smoke or make a shadow thinner to match shadows in the background imagery. This tool expands the Primatte medium polyhedron in the color region sampled.

Detail (+)

When this operational mode is selected, foreground detail will become less visible for the sampled pixel color (and all colors like it) in the amount of one Primatte increment. If there is still too much detail, another click using this operational mode tool will make more of it disappear. This can be used to remove smoke or wisps of hair from the composite. Sample where is visible and it will disappear. This is for moving color regions into the 100% background region. It can be used to nullify a **Detail (-)** step. This tool expands the Primatte small polyhedron in the color region sampled.

Detail (-)

When this operational mode is selected, foreground detail will become more visible for the sampled pixel color (and all colors like it) in the amount of one Primatte increment. If detail is still missing, another click using this operational mode tool will make detail more visible. This can be used to restore lost smoke or wisps of hair. Sample where the smoke or hair just disappears and it will return to visibility. This is for restoring color regions that were moved into the 100% background region. It may start to bring in background noise if shooting conditions were not ideal on the foreground image. This tool dents the Primatte small polyhedron in the color region sampled.

Current Color Chip

This shows the current color selected (or registered) by the **Fine Tuning** operational mode.

Fine Tuning Section

Fine Tuning sliders

When this operational mode is selected, the color of the sampled pixel within the Viewer window is registered as a reference color for fine tuning. It is displayed in the **Current Color Chip** in the **Actions** section. To perform the tuning operation, sample a color region on the image, select a Fine Tuning slider and move the slider to achieve the desired effect. See the **Fine Tuning Sliders** tool descriptions further on in this section for more details on slider selection.

Spill or L-poly slider (Spill Removal)

When in the **Fine Tuning Actions** mode, this **Spill** slider can be used to remove spill from the registered color region. After choosing the **Fine Tuning Actions** mode and registering a color region, this slider can be moved to remove spill from the registered color region. The more to the right the slider moves, the more spill will be removed. The more to the left the slider moves, the closer the color component of the selected region will be to the color in the original foreground image. If moving the slider all the way to the right does not remove all the spill, re-sample the color region and again move the slider to the right. These slider operations are additive. This result achieved by moving the slider to the right can also be achieved by clicking on the color region using the **Spill(-)** operational mode. This slider bulges the Primatte large polyhedron near the registered color region.

Transparency or M-poly slider (Adjust Transparency)

When in the **Fine Tuning Actions** mode, this **Transparency** slider can be used to make the matte more translucent in the registered color region. After choosing the **Fine Tuning Actions** mode and selected a color region, moving this slider to the right makes the registered color region more transparent. Moving the slider to the left makes the matte more opaque. If moving the slider all the way to the right does not make the color region translucent enough, re-sample the color region and again move the slider to the right. These slider operations are additive. This result achieved by moving the slider to the right can also be achieved by clicking on the color region using the **Matte(-)** operational mode. This slider bulges the Primatte medium polyhedron near the registered color region.

Detail or S-poly slider (Add/Restore Lost Detail)

When in the **Fine Tuning Actions** mode, this **Detail** slider can be used to restore lost detail. After choosing the **Fine Tuning Actions** mode and selected a color region, moving this slider to the left makes the registered color region more visible. Moving the slider to the left makes the color region less visible. If moving the slider all the way to the left does not make the color region visible enough, re-sample the color region and again move the slider to the left. These slider operations are additive. This result achieved by moving the slider to the left can also be achieved by clicking on the color region using the **Detail(-)** operational mode. This shrinks the small polyhedron (which contains all the blue or green background colors) and releases pixels that were close to the background color. The **S-poly** slider in the **Fine Tuning** mode is useful for restoring pixels that were lost because they were so similar to the background color. This slider dents the Primatte small polyhedron near the registered color region.

Spill Process Section

Complement/Solid/Defocus Spill Replacement

Allows the user to choose between the three methods of color spill replacement as covered in detail in Section 5. Spill Replacement Options and below.

no suppression

Replaces the spill color with the complement of the backing screen color.

complement

Replaces the spill color with the complement of the backing screen color.

solid color

Replaces the spill color with colors from a defocused version of the background image.

defocused background

Replaces the spill color with a 'user selected' solid color.

Replace color slider

When solid color is selected, this area allows the user to select a solid color to use to replace the spill. For all other spill replacement selections, this area is `grayed out' and not activated.

Defocus slider

When defocused background is selected, this area allows the user to adjust the amount of defocus applied to the background buffer image. For all other spill replacement selections, this area is `grayed out' and not activated.

Output Section

Output mode selector

These are the three formats for the output of the node.

composite

outputs the composite result of the Primatte node.

premultiplied

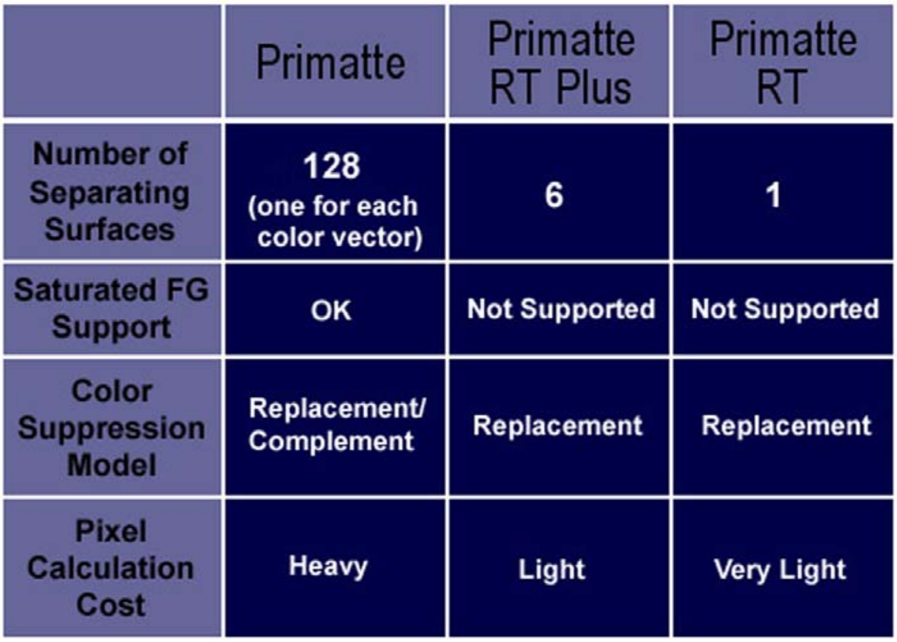
outputs the premultiplied result of the Primatte node.

unpremultiplied

outputs the unpremultiplied result of the Primatte node.

The Primatte Algorithm

There are three Primatte algorithms. Here is a chart that shows the main differences between them.



	Primatte	Primatte RT Plus	Primatte RT
Number of Separating Surfaces	128 (one for each color vector)	6	1
Saturated FG Support	OK	Not Supported	Not Supported
Color Suppression Model	Replacement/ Complement	Replacement	Replacement
Pixel Calculation Cost	Heavy	Light	Very Light

For a description of the Primatte algorithm, see "Explanation of How Primatte Works" below.

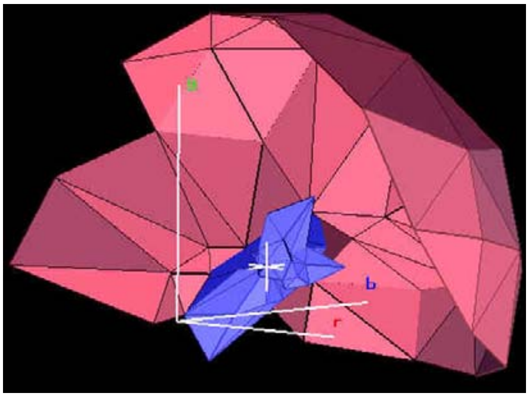
For a description of the Primatte RT+ algorithm, go to "Explanation of How Primatte RT+ works" on page 295.

For a description of the Primatte RT algorithm see "Explanation of How Primatte RT works" on page 296.

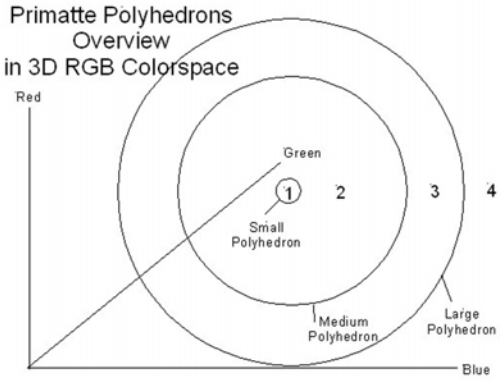
Explanation of How Primatte Works

The Primatte chromakey algorithm is a sophisticated method of color space segmentation that can be easily explained to help a user achieve maximum effectiveness with the tool. Basically Primatte segments all the colors in the foreground image into one of four separate categories. The result is a 'spill suppressed' foreground image and a matte which is used to apply the modified foreground to a suitable background.

Primatte works in 3D RGB color space. Here is a visual representation of the Primatte algorithm after an image has been processed.



By operating the Primatte interface, the user essentially creates three concentric, multi-faceted polyhedrons. These can be pictured as three globes (or polyhedrons or polys), one within the other, which share a common center point. The creation of these polyhedrons separates all possible foreground colors into one of four regions; inside the small polyhedron **(1)**, between the small and medium polyhedrons **(2)**, between the medium and the large polyhedrons **(3)** and outside the large polyhedron **(4)**.

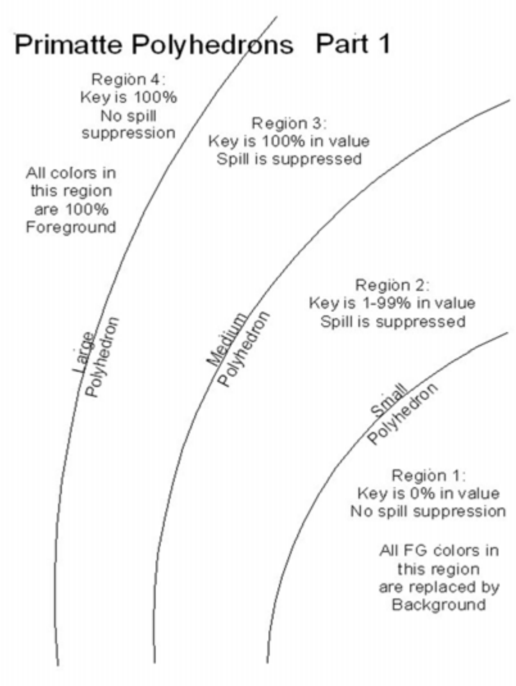

Primatte Polyhedrons
Overview
in 3D RGB Colorspace
Red
Green
1
2
3
4
Small
Polyhedron
Medium
Polyhedron
Large
Polyhedron
Blue

The four regions created are described as follows:

Region 1 (inside the small polyhedron) - This region contains all of the foreground image colors that are considered 100% background. These are the green or blue or whatever colors that were used as the backing color of the foreground image.

Region 2 (between the small and medium polyhedrons) - This region contains all the foreground colors that are at the edges of the foreground object(s), in glass, glass reflections, shadows, sheets of water and other transparent and semi-transparent color regions. These color regions also have spill suppression applied to them to remove color spill from the backing screen.

Region 3 (between the medium and large polyhedrons) - This region contains all the foreground image colors that are 100% foreground but have spill suppression applied to them to remove color spill from the backing screen. Otherwise they are 100% solid foreground colors.


Primatte Polyhedrons Part 1
Region 4:
Key is 100%
No spill
suppression
All colors in
this region
are 100%
Foreground
Region 3:
Key is 100% in value
Spill is suppressed
Region 2:
Key is 1-99% in value
Spill is suppressed
Large
Polyhedron
Medium
Polyhedron
Small
Polyhedron
Region 1:
Key is 0% in value
No spill suppression
All FG colors in
this region
are replaced by
Background

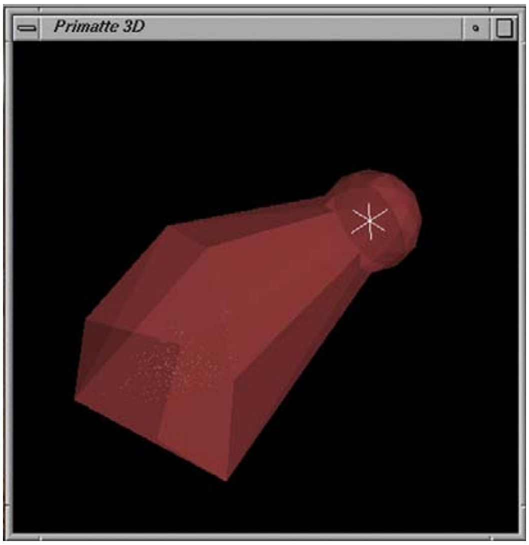
Region 4 (outside the large polyhedron) - This region contains all the 100% foreground image colors that are not modified from the original foreground image. There is no spill suppression applied to these colors.

In the first step in using Primatte (**Select Background Color**), the user is asked to indicate the backing color on the original foreground image. The sample should usually be taken from a 'medium shaded' area near the foreground object. By 'medium shaded' area, it is meant that if green is the backing color and the green area of the foreground image has many shades of green ranging from very pale green to almost black, a shade of green in-between these extreme ranges should be chosen. If good results are not obtained using this sample, Primatte should be reset and another sample taken using a slightly darker or lighter shade of green. The first sample of Primatte often determines the final result as the center point of all three polyhedrons is created based on this first sample.

A single pixel may be selected or a range of pixels (snail trail or rectangular sample). If a range of pixels is taken, the sample will be averaged to get a single color sample. This single pixel or averaged color sample then becomes the center of the small polyhedron. A few other shades around that color are included in the original small polyhedron.

Note *It is recommended that a single pixel be selected as the first sample as you then have some idea where the center point of the polyhedrons is located. If a box sample or a long snail trail sample is made. You can only guess at the average color that ends up being the center point. You can get an idea how this sample affects the algorithm by resetting the Primatte plug-in, going to the Matte View and clicking around on the green or blue screen area while in the Select Background Color Operation Mode. You can immediately see the results of the initial settings of the polyhedrons in this way.*

After making a sample of the backing screen color in the first step, the result is a small golf ball-shaped poly as shown in the following image.



The second step in using Primatte is to clean up the backing color area by adding additional shades of green or blue to the small poly. This second step (**Clean Background Noise**) is usually executed while viewing the black and white **Matte View**.

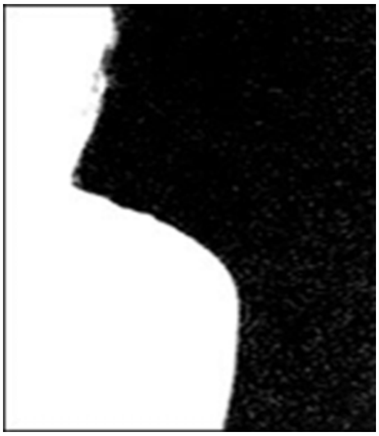


Figure 7.23: Before background noise removal.

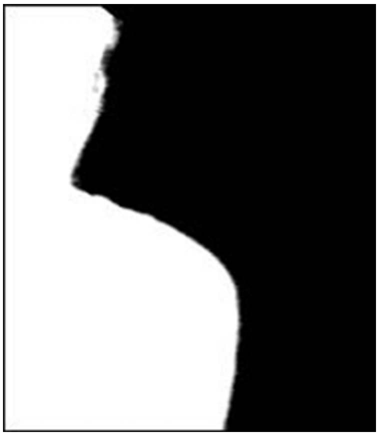
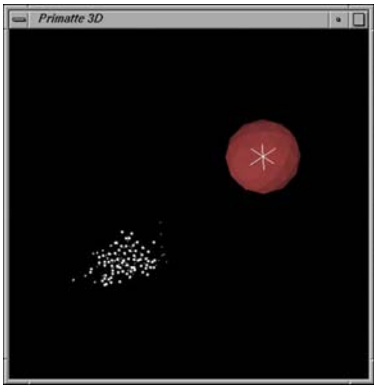
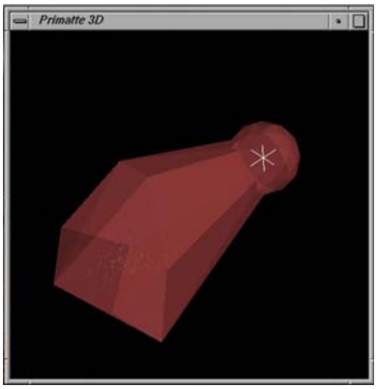


Figure 7.24: After background noise removal.

While in the **Clean Bg Noise** sampling mode, the user samples the white milky regions as shown in the left-hand image above. As the user samples these regions, they turn to black as shown in the right-hand image above.

What is happening in the Primatte algorithm is that these new shades of green (the white milky areas) are added to the small poly where all the shades of green or blue are moved. The next two images show the new pixels sampled (white dots) in relation to the small poly and the image next to it shows how the small poly extends outward to encompass the newly sampled colors into the small poly.





The advantage of this technique is that the polyhedron distorts to enclose only the shades of green that are in the backing screen. Other shades of green around these colors are left undisturbed in the foreground. Other chromakeyers expand from a golf ball-sized shape to a baseball to a basketball to a beach ball. Since it expands in all directions, many shades of green are relegated to 100% background making it hard to get good edges around the foreground objects.

Now that the user has created a small polyhedron, he must shape the medium and large polys. A default medium and large poly are both automatically created and are then modified based on the next couple of Primatte operations. The third Primatte step (**Clean Foreground Noise**) is to sample and eliminate gray areas in the 100% foreground area of the image.

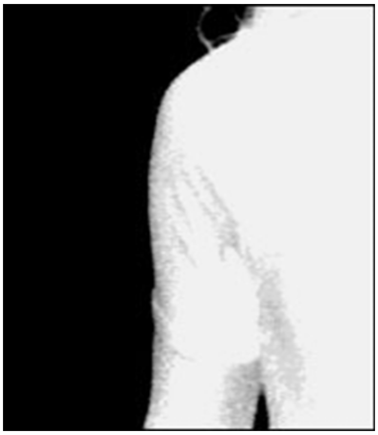


Figure 7.25: Before and after foreground noise removal.

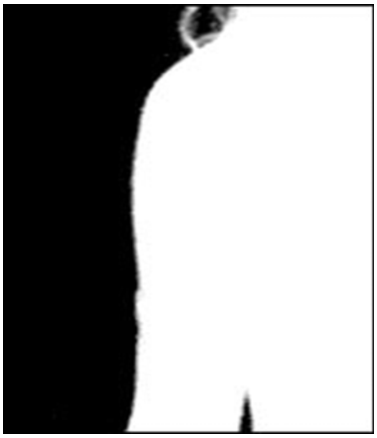
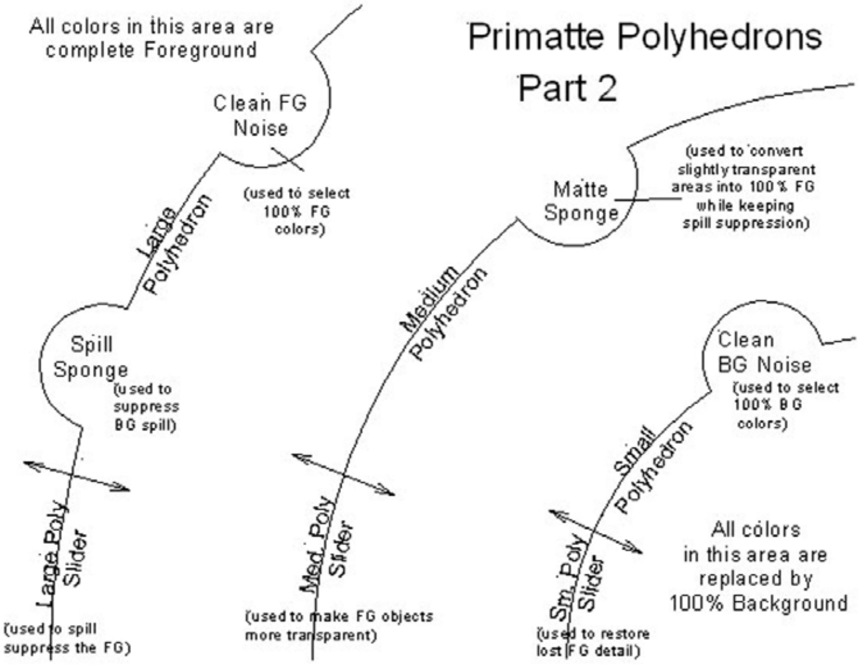


Figure 7.26: Before and after foreground noise removal.

Again, the user makes several samples on the dark, grayish areas on the foreground object until it is solid white in color. Primatte is shaping the large polyhedron with each color region that is sampled. Care should be taken in both this and the previous steps to not sample too close to the edges of the foreground object. Getting too close to the foreground object's edges will result in hard edges around the foreground object. Primatte uses these samples to modify and shape the medium and large polys to the desired shape. At this point, the matte or key has been created and would allow the foreground objects to be composited into a new background image.

If the user changes the display mode from the black and white **Matte View** to the color **Composite View**, there is usually 'color spill' on the edges (and sometimes the center) of the foreground objects. When on the edges of the foreground object, this spill comes from where the edges of the foreground object blended into the backing color. If it is on the center of the foreground object, it usually results from reflected color from the backing screen. The next Primatte step, either **Spill Sponge**, **Fine Tuning** or **Spill(-)**, can now be used to eliminate this spill color.

Let's take a look at what is happening in the Primatte algorithm while this next step is performed. Here is what the various tools in Primatte do to the Polyhedrons when they are used:


Primatte Polyhedrons
Part 2
All colors in this area are complete Foreground
Clean FG Noise
(used to select 100% FG colors)
Large Polyhedron
Spill Sponge
(used to suppress BG spill)
Large Poly Slider
(used to spill suppress the FG)
Matte Sponge
(used to convert slightly transparent areas into 100% FG while keeping spill suppression)
Medium Polyhedron
Med. Poly Slider
(used to make FG objects more transparent)
Clean BG Noise
(used to select 100% BG colors)
Small Polyhedron
Sm. Poly Slider
(used to restore lost FG detail)
All colors in this area are replaced by 100% Background

As you can see above, the **Spill Sponge** bulges the large polyhedron in the color region specified. A color region is specified by clicking on the image in a particular area with spill present. For example, if the user clicks on some spill on the cheek of a foreground person, Primatte goes to the section of the large polyhedron closest to that particular flesh tone and bulges the polyhedron there. As a result, the flesh tones move from outside the large poly to in-between the medium and large polys. This is **Region 3** and, if you remember, is 100% foreground with spill suppression. As a result of the suppression, the spill is removed from that cheek color and all other shades of that color on the foreground. The user would then continue to sample areas of the image where spill exists and each sample would remove spill from another color region.

When all spill has been removed, the user should have a final composite. As a last step, he should go back to the **Matte View** and make sure that gray, transparent areas have not appeared in the foreground area. If there are any, the **Matte Sponge Operation Mode** should be selected and those gray pixels are sampled until they have all turned white again.

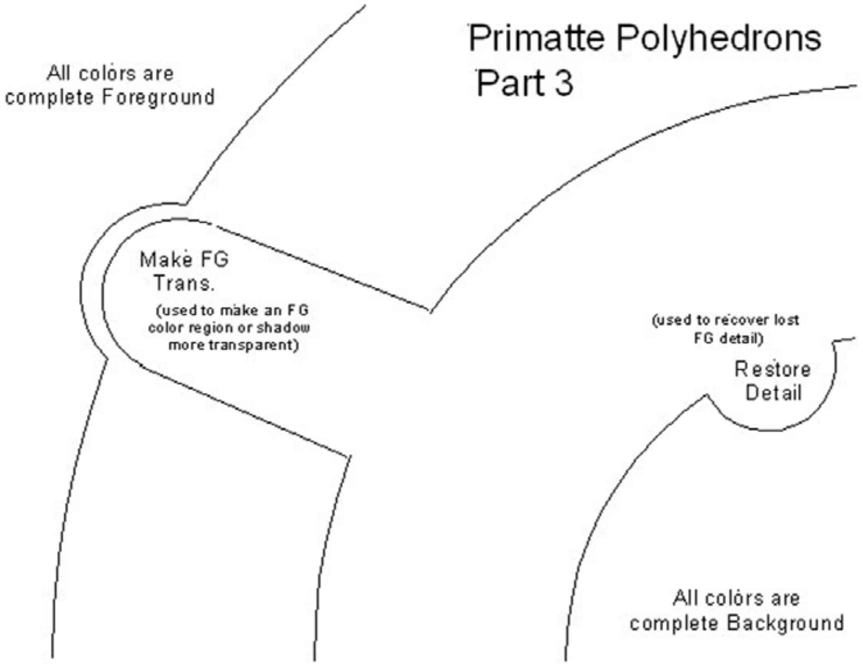
The **Matte Sponge** and **Spill Sponge** tools bulge or dent the polyhedrons a pre-selected amount. If the desired results are not achieved or the results are too extreme for the image, a manual method can be applied. The user should choose the **Fine Tuning** sliders, select a color region of interest and then move the appropriate slider to get the desired results.

For example, to remove spill, select a region of the composite image with spill on it. Move the spill or large poly slider to the right a little bit, the large poly will bulge and the spill should disappear. Move it a little more, if necessary. Moving this slider to the right removes spill (moves the colors from outside the large poly to between the medium and large polyhedrons) and moving it to the left, dents the large poly and moves that color region to outside the large poly.

If the user samples a foreground object shadow and then moves the **Matte** or medium poly slider to the right, the shadow will become more transparent. This is useful for matching composited shadows to shadows on the plate photography. It can also be used to make clouds or smoke more transparent.

If some foreground detail disappears during the composite, the user can select where the detail should be and then move the detail or small poly slider to the left. This dents the small poly in that color region and releases the detail pixels from the small poly into the visible region between the small and medium polyhedrons.

The **Spill Sponge** and **Matte Sponge** tools are 'shortcut tools' that automatically move the sliders a pre-selected amount as a timesaving step for the user. Other 'shortcut tools' include the **Make Foreground Trans.** tool and the **Restore Detail** tool.


Primatte Polyhedrons
Part 3
All colors are
complete Foreground
Make FG
Trans.
(used to make an FG
color region or shadow
more transparent)
(used to recover lost
FG detail)
Restore
Detail
All colors are
complete Background

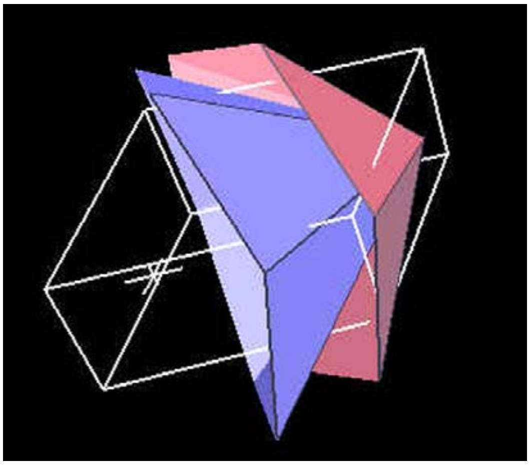
These 'shortcut tools' are one-step operations where the user clicks on a color region of interest and Primatte performs a pre-calculated operation. Hopefully, most operations using Primatte would only require these tools, but the manual operation of the sliders is always an option.

The **Spill(-)** tool bulges the large poly a small amount incrementally in the color region that is clicked on and the **Spill(+)** tool dents it a small amount with each click. The **Matte(-)** and **Matte(+)** tools do the same to the medium poly and the **Detail(-)** and **Detail(+)** do it to the small poly.

Explanation of How Primatte RT+ works

The **Primatte RT+** algorithm differs from the Primatte algorithm in that it has a six surface color separator instead of the 127-faceted polyhedrons. This makes the **Primatte RT+** algorithm much simpler and, therefore, faster to calculate. The results and performance of **Primatte RT+** falls in between the **Primatte** and **Primatte RT** options. Where the **Primatte RT+** algorithm might not work well is with less saturated backing screen colors and it also does not support the **Complement Color** spill suppression method (which is the spill suppression method that delivers the best detail). For a well-lit and photographed image or clip, this algorithm will produce good results and render quickly.

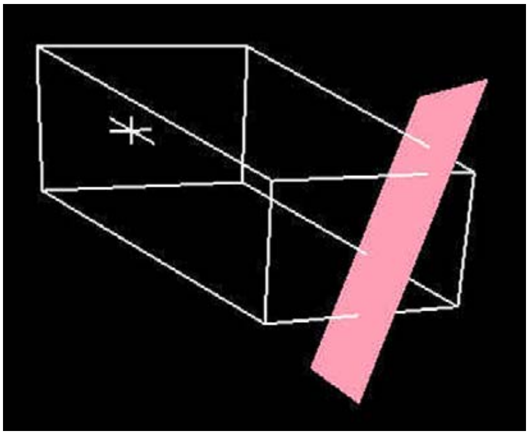
Here is what a visual representation of the Primatte RT algorithm looks like after an image has been processed:



Explanation of How Primatte RT works

Primatte RT is the simplest algorithm and, therefore, the fastest. It uses only a single planar surface to separate the 3D RGB colorspace and, as a result, does not have the ability to separate out the foreground from the backing screen as carefully as the above Primatte algorithm. Like the **Primatte RT+** algorithm, **Primatte RT** might not work well with less saturated backing screen colors and it too does not support the **Complement Color** spill suppression method (which is the spill suppression method that delivers the best detail). For a well-lit and photographed image or clip, this algorithm will produce good results and render very quickly.

Here is what a visual representation of the **Primatte RT** algorithm looks like after an image has been processed:



Contact Details

Main Office

IMAGICA Corp. of America, 1840 Century Park East, #750, Los Angeles, CA, USA 90067

Primatte Office

IMAGICA Corp. of America, 3113 Woodleigh Lane, Cameron Park, CA 95682. Phone: 1-530-677-9980, FAX: 1-530-677-9981, Cell: 1-530-613-3212, E-mail: sgross@imagica-la.com, Website: http://primatte.com

Proprietary Notices

Primatte is distributed and licensed by IMAGICA Corp. of America, Los Angeles, CA, USA. Primatte was developed by IMAGICA Corp., Tokyo, Japan. Primatte is a trademark of IMAGICA Corp., Tokyo, Japan.

8 KEYING WITH KEYLIGHT

This section explains how to use the blue/green screen keyer, Keylight, in Nuke.

Quick Key

Consider this shot from The Saint, pictures courtesy of CFC and Paramount British Pictures Ltd.

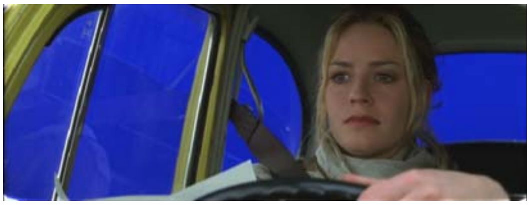


Figure 8.1: Blue screen.

Figure 8.1 is the blue screen foreground that should be composited over the background shown in Figure 8.2.

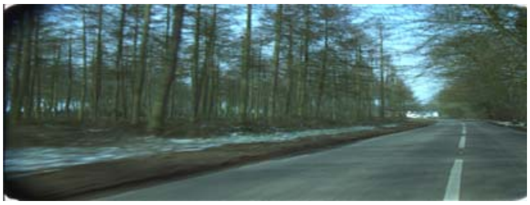
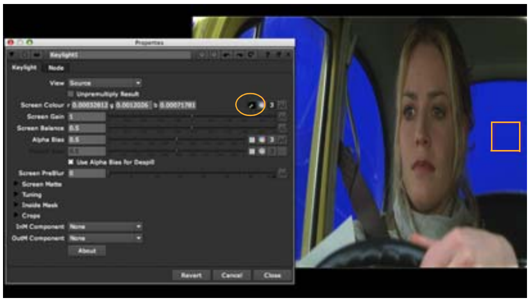


Figure 8.2: Background.

1. Start Nuke and read in both images. From the **Keyer** menu, apply **Keylight** and attach a Viewer.
2. Click the color swatch next to **Screen Color** to activate the eye dropper. In the Viewer, **Ctrl+Shift+Alt**+click (Mac users **Cmd+Shift+Alt**+click) and drag a rectangular area over the blue pixels as shown in Figure 8.3.



View: Source
Screen Gain: 1
Screen Balance: 0.5
Alpha Bias: 0.5
☒ Use Alpha Bias for Despill
Screen PreBlur: 0
InM Component: None
OutM Component: None

Figure 8.3: Pick the **Screen Color**.

Picking the screen color also sets the **Screen Balance**.

3. That's it. In many cases, this is all you will need to do to perform a key, since selecting the screen color creates a screen matte and despills the foreground.
4. Switch output from **Final Result** to **Composite** to see the foreground keyed over the background. The final composite is shown in Figure 8.4.

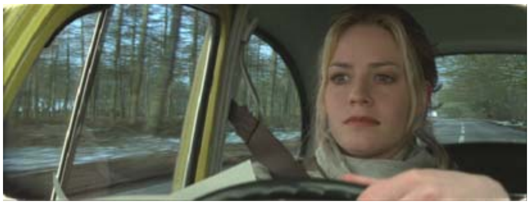


Figure 8.4: Final composite.

Picking the screen color may be enough for a lot of keys, but there are many more tools within Nuke that can be used to tackle more complicated shots. These are described later in this chapter.

Basic Keying

The following section describes the parameters you need to do basic keying. This will give you enough to tackle most simple keys. A discussion of advanced parameters to fine tune keys and tackle complex keys can be found on page 301.

Picking the Screen Color

The screen color is probably the most important parameter and you should always pick the screen color before doing anything else. It should be set to the color of the green or blue curtain behind the foreground object. Pick the screen color directly from the image by **Ctrl+Shift+Alt** dragging (**Cmd+Shift+Alt**+dragging on a Mac) a rectangle over the blue pixels. The average value of the pixels selected is used.

Tip *If you press* **Alt** *when sampling a color, Nuke always samples the source image regardless of what you're looking at. This means that you can pick the blue screen color even if you are viewing the matte, status or composite.*

Tip *Picking different shades of blue from the screen color can give quite different results. It's worth picking from different parts of the screen to get the best result.*

Picking the **Screen Color** creates the screen matte used to composite the foreground over the background. It also sets the **Screen Balance** (if this has not already been set manually) and despills the foreground.

Screen Matte

Setting the screen color will pull a key, or in other words, create a matte - the **Screen Matte**. Setting the screen color will also despill the foreground, although you can also use the **Despill Bias** to remove more spill. In some cases this is enough to get a decent key. For more information on **Screen Color** see page 299.

Figure 8.5 shows a well-lit blue screen behind an actor.

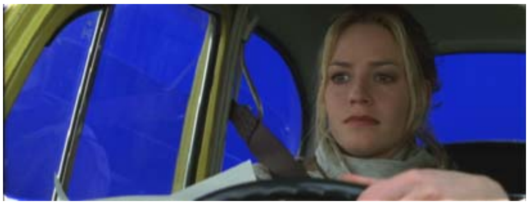


Figure 8.5: Blue Screen.

You should note that repeatedly picking colors does not add to previous selections and key more of the image with each click. To key more of the image, try picking different shades of blue then use the screen strength parameter. See "Keying More" on page 301.

Viewing the Key

After picking the screen color, you have created a matte (the screen matte) and despilled the foreground. The result can be displayed in a number of different ways using the **View** control. You can output the final composite of the foreground over the background as an rgba, or you can output the premultiplied or unpremultiplied foreground for compositing elsewhere in the tree. The screen matte and the status view are the other two options which are useful in fine tuning the key rather than as an output image in their own right.

The **Status** is one of the options in the **View** menu and shows an exaggerated view of the key so that you can make a more informed decision when tuning the key. Figure 8.7 shows the **Status** display after the screen color has been picked from the image shown in Figure 8.6.

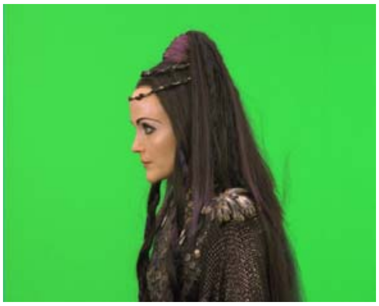


Figure 8.6: Green screen.

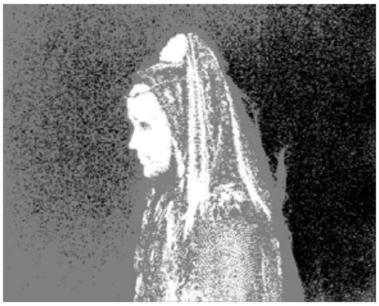


Figure 8.7: Status.

Three colors are displayed:

- Black pixels show areas that will be pure background in the final composite.
- White pixels show areas that will be pure foreground.
- Gray pixels will be a blend of foreground and background pixels in the final composite. You need gray pixels around the edge of the foreground to get a good key at the foreground edge. Pixels that are a blend between the foreground and background are shown in just one shade of gray. This is done to highlight potential problems with the key. These gray pixels may represent a foreground/background blend of 50/50 or 99/1. No distinction is made as to this ratio.

You may occasionally see other colors in the **Status** view and these are covered under "Status" on page 302.

Keying More

To improve the key by firming up the foreground so the background doesn't show through, you should adjust the **Clip White** parameter. To key more of the foreground so that the background is clearer, you should use the **Clip Black** parameter. Look at the **Screen Matte** and the **Composite** while you're doing this. Don't overdo either of these or the edges between foreground and background will become hard.

Advanced Keying

The following section describes how Keylight works under the hood as well as the parameters you need to fine tune keys and get the most out of Keylight. Basic parameters covered in the previous chapter may also be covered here in more detail.

Under the Hood

Keylight is a 'color difference keyer', which means that for it to figure out a key, it compares every pixel in the image against a single color, known here as the **Screen Color**.

View

The **View** parameter allows Keylight to render the final composite of the foreground over the background, or the foreground RGBA for compositing further down the tree. Two options, **Screen Matte** and **Status**, are for viewing the key rather than an output. The options are:

- **Source** - shows the blue/green screen foreground.
- **Source Alpha** - shows the alpha channel on the foreground input.
- **Screen Matte** - this is the matte created from picking the **Screen Color**. It does not include any inside or outside masks.
- **Inside Mask** - shows the inside input. This is used to firm up the foreground matte to stop print through.
- **Outside Mask** - shows the outside input. The outside mask is used as a garbage mask to reveal the background.
- **Combined Matte** - the screen matte, inside mask, and outside masks added together.
- **Status** - this renders an exaggerated view of the key so that minor problems are shown clearly. See "Status" on page 302.
- **Intermediate Result** - use this option on shots that can only be keyed using several different keys on different parts of the image (multipass keying). This renders the original source image with the **Screen Matte** generated in this Keylight node. In Keylight nodes down the tree, you should set the **Source Alpha** in the **Inside Mask** folder to **Add To Inside Mask**.
- **Final Result** - this creates a premultiplied RGBA foreground that can be composited later. There's an **Unpremultiply Result** toggle you can use if you wish.
- **Composite** - this renders the foreground composited over the background using all mattes, spill and color corrections.

Status

Status is one of the options in the View menu and shows an exaggerated view of the key so that you can make a more informed decision when fine tuning the composite. Figure 8.7 shows the Status after the screen color has been picked from the image shown in Figure 8.6 on page 301.

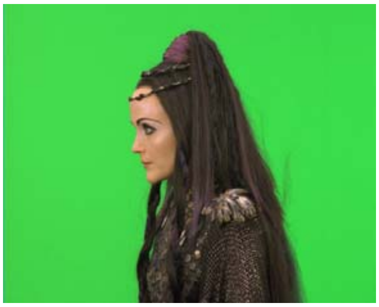


Figure 8.8: Green screen.

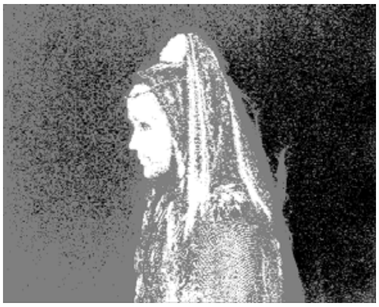


Figure 8.9: Status.

Three colors are displayed:

- Black pixels represent pure background in the final composite.
- White pixels are pure foreground.
- Gray pixels are a blend of the foreground and background pixels. The gray is just one color to highlight any areas that are not pure foreground or background. Gray pixels do not mean the key is poor - the final composite may be fine.

You may occasionally see other colors in the **Status** view. Figure 8.10 shows black, white, gray, and green pixels.

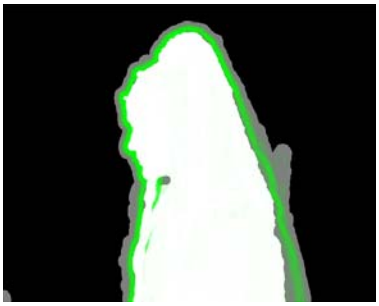


Figure 8.10: Status showing processing of the alpha channel.

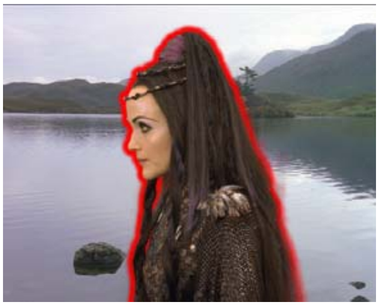


Figure 8.11: Composite showing **Screen Replace Color**.

- Green pixels are a warning. They show you the parts of the alpha that have changed through processing the alpha channel (clipped, softened, or eroded). These areas have had the correct amount of spill removed, but the alpha has subsequently changed and the composite may no longer look right. This can be corrected using the **Screen Replace Color** to put back color in these areas. Figure 8.11 is an extreme example to illustrate the point. The **Screen Replace Color** has been set to pure red and you can see that this mirrors the green pixels in the **Status** view.

Similarly, you may see blue pixels in the **Status**.

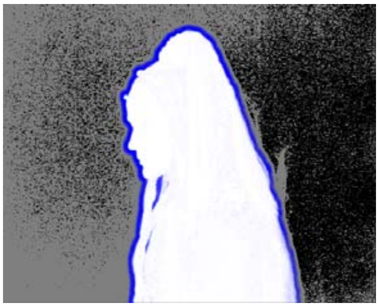


Figure 8.12: Status showing how the inside matte will affect the foreground.

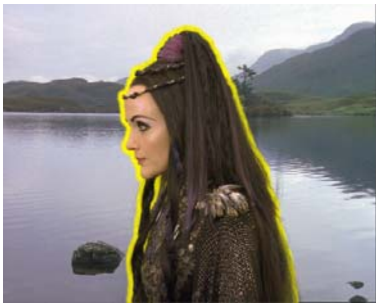


Figure 8.13: Composite showing the inside replace color.

- Blue pixels represent processed pixels in the **Inside Mask** that affect the despill of the foreground. The **Inside Replace Color** will be used to modify these pixels. Another extreme example is shown in Figure 8.13. The **Inside Replace Color** is set to pure yellow and the **Inside Replace** is **Hard Color**.
- You may also see dark red pixels in the **Status**. Red pixels indicate areas where an outside mask has been used to reduce the transparency of the image.

Screen Color

The screen color represents the color of the pure blue (or green) screen. The first thing you should do when pulling a key is pick the **Screen Color**.

Note *If you press **Alt** when sampling a color, Nuke always samples the source image regardless of what you're looking at. This means that you can pick the blue screen color even if you are viewing the matte, status or composite.*

Picking the **Screen Color** creates the screen matte used to composite the foreground over the background. It also sets the **Screen Balance** and despills the foreground.

The **Screen Color** is a single color. It has a primary component, blue or green, and that has a saturation. Once the screen color has been picked, Keylight analyzes all the pixels in the image and compares the saturation of the primary component in each of these pixels with the corresponding saturation of the screen color. Keylight uses this comparison to do two things.

1. It calculates the transparency of that pixel and puts it in the alpha channel.
2. It removes the screen color from the pixel, a process known as despilling.

Tip *It's worth sampling a selection of screen (blue or green) colors and viewing the result. Picking different colors will give different results.*

Background Pixel

If the saturation of the pixel in the image is as strong or greater than the screen color, then it'll be a pixel from the blue screen background, and that pixel will be set to completely transparent and black. See Figure 8.14.

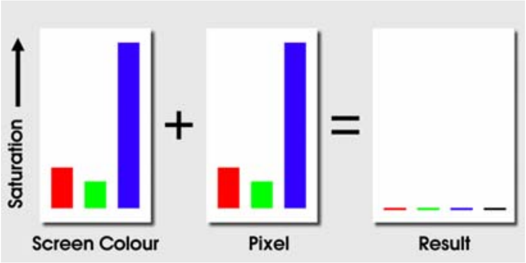


Figure 8.14: Blue screen pixel set alpha to zero.

Edge Pixel

If the saturation of the pixel is less than the screen color, then it'll be the edge of the foreground object, and we subtract some of the screen color from the pixel (despilling) and set the image to semi-opaque. See Figure 8.15.

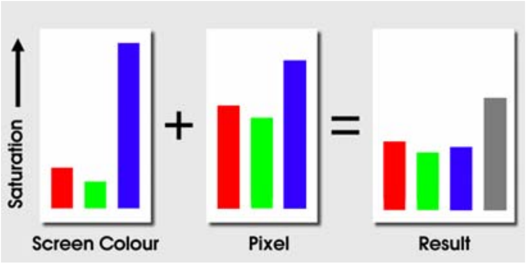


Figure 8.15: Edge pixel gives partial alpha.

Foreground pixel

If the primary component in the pixel is not the same as the primary component of the screen color, we have a foreground pixel, and the alpha is set to completely opaque. The pixel color is not modified. See Figure 8.16.

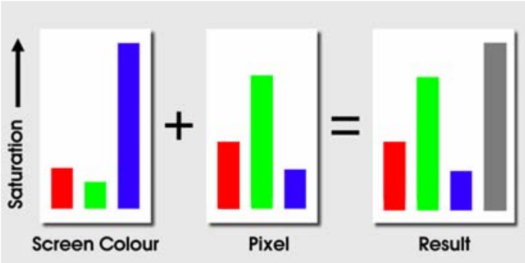


Figure 8.16: Foreground pixel gives full alpha.

Note *You should note that the **Screen Color** is a single color. You are not picking lots of colors that are keyed out.*

Biasing

What's biasing all about? Biasing in Keylight was originally developed for a shot in the motion picture "Executive Decision". The foreground consisted of reddish browns, but a combination of factors led to the 'green screen' being lit so that its primary component was actually slightly red.

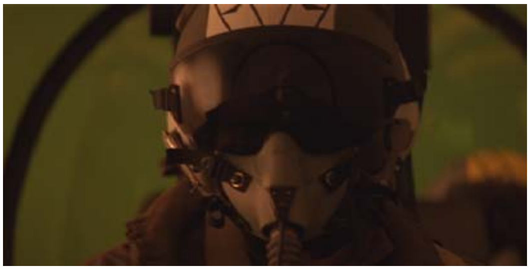


Figure 8.17: Is this the worst green screen you've ever seen?

So what happens when we pick the screen color? Well because the screen was 'red', as is the foreground, our pilot ends up being keyed out as shown in Figure 8.18.

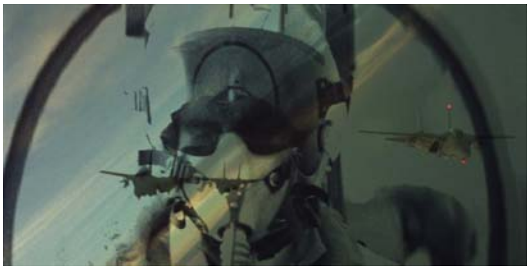


Figure 8.18: Default key showing the transparency in the foreground as a result of picking the "red" screen color.

Not a great result, I'm sure you'll agree, and much pressure was applied to the lowly programmers to get around the problem.

A work around to this is to manually color correct the image so that the background is properly green, pull the key from this corrected image, then 'un-correct' the result of that so that the foreground colors match the original. A corrected image would look something like the one shown in Figure 8.19. The green screen is now strongly green and distinct from the foreground colors. Notice also the red cast on the pilots mask has been removed and turned into a neutral gray.

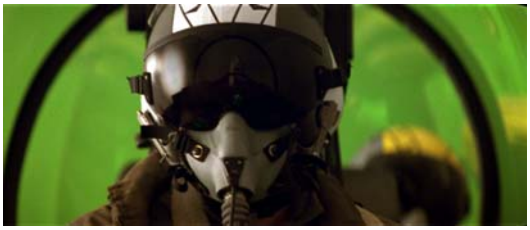


Figure 8.19: Color corrected image that would give a better key.

This is effectively how the Keylight developers got around the problem. They introduced the concept of a 'bias' color, which is a color cast that is removed from the source image and screen color, then a key is pulled from this modified image, then the color cast is put back. In essence, this automates the work around described above, however, it is done in a way that does not slow Keylight down at all.

For our Executive Decision shot, an appropriate color is the red cast on the pilot's mask in the source footage. Setting our bias to this now gives us the far better result as shown in Figure 8.20.

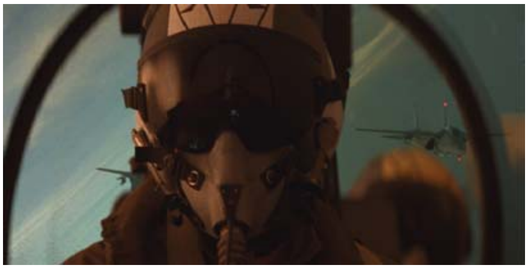


Figure 8.20: Final Key, with the Bias Color Set to the Value of the Pilot's Mask.

The bias colors in everyday use

It also turns out that the bias color is actually useful for situations without strong casts, typically where there is some color spill around the edge of keys. By setting the biases to the main color that occurs near the edge of the foreground (typically flesh tones or hair tones), you allow Keylight to better discriminate between foreground and background.

Picking a bias color

To pick a bias color, click the color swatch next to **Alpha Bias** to activate an eye dropper and **Ctrl+Shift+Alt**+drag (Mac users **Cmd+Shift+Alt**+drag) a box over the image foreground. The average color under the box will be used for the bias you have chosen.

Note *If you press **Alt** when sampling a color, Nuke always samples the source image regardless of what you're looking at. For instance, you may be looking at the blue screen keyed over the background but you will be picking colors from the Source image.*

Why are there two bias colors?

Remember that Keylight does two things, calculates a transparency and removes the screen color from the foreground. By default, one bias color, the 'Alpha Bias, is used for both operations. This works fine in most situations, for example, the Executive Decision shot above.

However, sometimes you can pick a bias that gives a great alpha, but performs a poor despill, and another bias that gives a great despill, but a poor alpha. Consider the blue screen from the TV series Merlin, courtesy of CFC Framestore shown below in Figure 8.21.

We pick the strong blue of the background without choosing an alpha bias, and end up with the lovely alpha shown in Figure 8.22, but the despill resulting from this key is poor as shown in Figure 8.23.

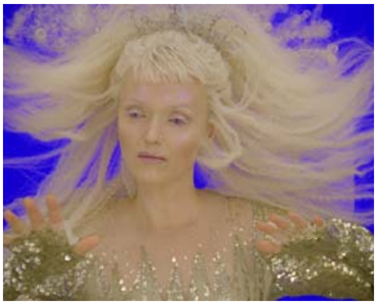


Figure 8.21: Merlin blue screen.

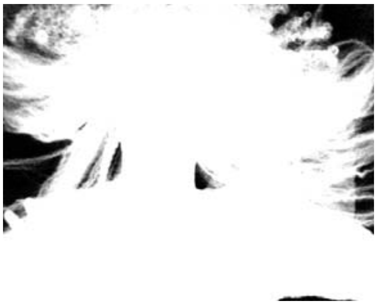


Figure 8.22: Nice Alpha.

We can pick an alpha bias to get a better despill, but this destroys our nice alpha. The way around this is to turn off the **Use Alpha Bias for Despill**, which gives you a separate bias factor to use solely for despill calculations. If you then pick the **Despill Bias** to be something from Miranda Richardson's hair or skin tone, you will keep the nice alpha, and get a good despill as well (Figure 8.24).

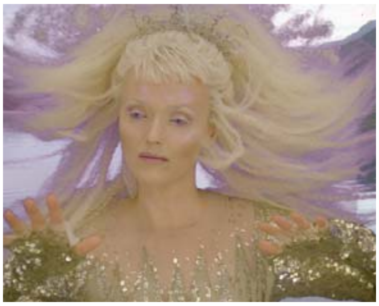


Figure 8.23: Poor despill.

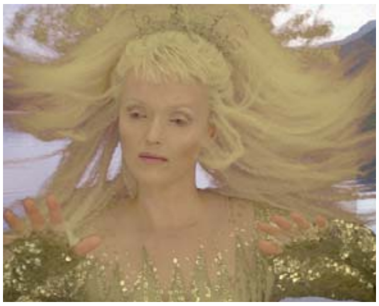


Figure 8.24: Final Key, Using Separate **Despill** and **Alpha Biases**.

Clip Black and White

The clip levels are adjusted using two parameters - **Clip Black** and **Clip White**. Any alpha value at or below **Clip Black** will be set to zero and any alpha value at or above **Clip White** will be set to 1. Figure 8.25 shows the original alpha of an image and Figure 8.26 shows the result of clipping it.

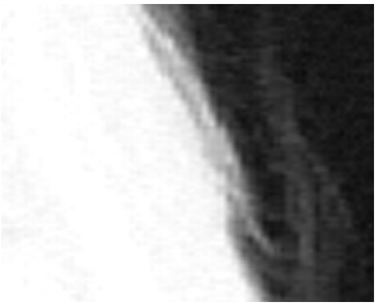


Figure 8.25: Clip Black = 0.

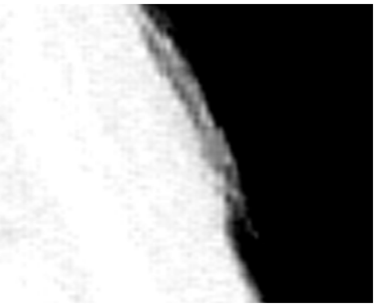


Figure 8.26: Clip Black = 0.5.

Notice how the gray areas in the black background have been reduced and that the gray edges have hardened up considerably. When compositing, the **Clip Black** control can be used to improve the background image if parts of the foreground are showing through. The **Clip White** control, on the other hand, can be used to firm up the center of the matte, making it less transparent to the background.

Note *If you choose to use **Clip Black** and **Clip White**, you need to be really careful that you don't destroy the edges on your foreground. It is possible to use **Clip Rollback** to compensate for this.*

Screen Gain

The screen gain controls how much of the screen color is removed to make the screen matte. Increasing this value will key more. Figure 8.27 shows the **Status** after picking the **Screen Color**.

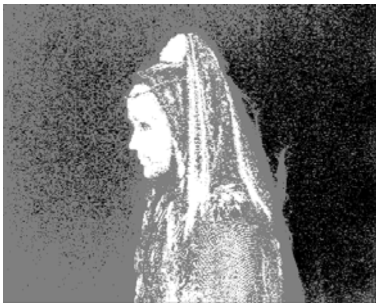


Figure 8.27: **Status** after picking the **Screen Color**.

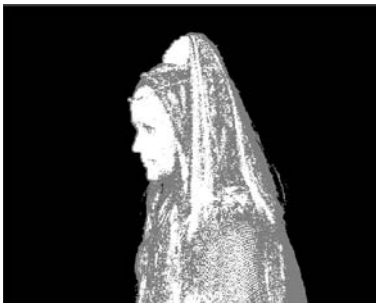


Figure 8.28: **Status** showing the increase in **Screen Gain**.

You can clearly see that parts of the background are gray where they should be black. When composited, you may see faint pixels from the foreground where you should be seeing pure background. Increasing the screen gain will fix this, as shown in Figure 8.28, but increasing it too much will destroy your good work. Like many keying parameters it's a balance - not too much, not too little. Increasing the screen gain too much will lead to, the background showing through the foreground and edge detail will be destroyed. Figure 8.30 shows this quite well.

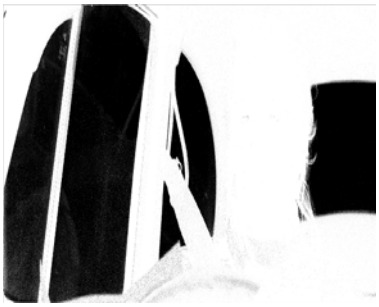


Figure 8.29: **Screen Gain** = 1.05 giving a good Screen Matte.

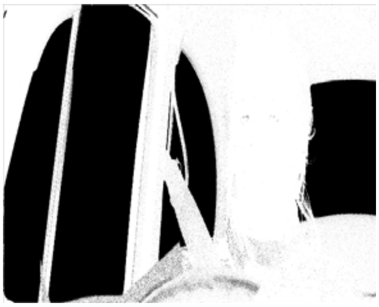


Figure 8.30: **Screen Gain** = 1.50 giving background show through and over eroded edges.

Note the steering wheel is black when it should be white. If you look at the composite, you will see the background showing through here. Also, some of the fine hair detail on the actor, visible in Figure 8.29, has been eroded in Figure 8.30.

Screen Balance

The **Screen Balance** is set automatically after picking the **Screen Color**.

Saturation is measured by comparing the intensity of the primary component against a weighted average of the two other components. This is where the Screen Balance control comes in. A balance of 1 means that the saturation will be measured against the smallest of the other two components in the screen color.

A balance of 0 means that the saturation will be measured against the larger of the other two components. A balance of 0.5 will measure the saturation from the average of the other two components.

The appropriate balance point for each image sequence you key will be different depending on the colors in that image. Generally speaking, blue screens tend to work best with a balance of around 0.95 and green screens with a balance of around 0.5. These values are selected automatically the first time you pick the screen color. If the key is not working too well with these settings, try setting the balance to about 0.05, 0.5 and 0.95 and see what works best.

PreBlur

Some shots can be improved by softening the foreground image that is used to generate the key. The original image is then used in the composite and color corrections. The **Screen PreBlur** parameter is used to do this. DV footage or grainy shots may benefit from subtle use of this control.

Tuning

Keylight creates the screen matte after the screen color has been picked. You can make fine adjustments to this matte using the **Gain** controls. Increasing the gain controls makes the screen matte more transparent by increasing the amount of screen color showing through the matte. This tends to tint the edges the opposite of the screen color (for blue screens, edges become yellow). Decreasing the gain makes the main matte more opaque by reducing the amount of screen color showing through the matte.

The matte can be adjusted independently in the shadows, midtones, and highlights, giving more control than the clipping levels.

The level of the midtones can be adjusted too. For example, if you are working on a dark shot you may want to set the midtone level to a dark gray to make the gain controls differentiate between tones that would otherwise all be considered shadows.

Screen Processing

Once you have picked the screen color and got the screen matte, you may wish to process this matte using the parameters in the **Screen Matte** group. The matte can be adjusted using clipping levels; it can be eroded or grown, despotted, and softened.

Two stage keying

Consider this example. Having applied Keylight and picked the screen color, you have good edges to your matte but the background is showing through the foreground. You could fix this by tweaking the **Clip White**, but in doing so it ruins your edges. One way round this is a two stage key (another way is using **Clip Rollback**). In the first key, you process the screen matte using the clipping levels to give a harsh black and white matte, then soften and erode it. Switch **View** to **Intermediate Result** to output the original green screen with the eroded matte as an RGBA. Then, use this as the input to another Keylight node. In this second node, pick the screen color to give good edges but with transparent foreground. Don't process this matte, instead use the input alpha channel to fix the transparent foreground. Just set **Source Alpha** in the **Inside Mask** folder to **Add To Inside Mask**.

Clip Rollback

Pulling a screen matte (Figure 8.31) will typically produce lots of transparency (gray) in the matte at the edges. This is good since this is what you need to key hair well. You may also get transparency in the foreground as shown in Figure 8.32. This is bad as your subject will appear slightly see-through, and this should be corrected.

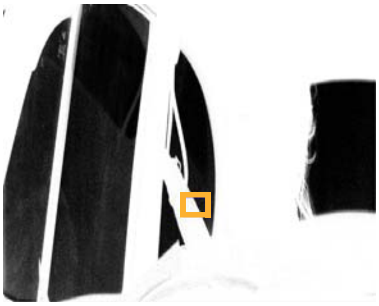


Figure 8.31: Screen matte highlighting the close up view as shown in Figure 8.32.

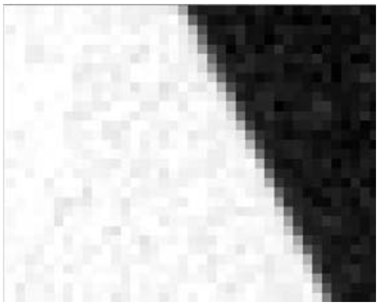


Figure 8.32: Close up screen matte showing unwanted (gray) transparency in the (white) foreground.

You can do this by connecting a matte into the third (**InM**) input, or you can use the **Clip White** parameter to turn these gray pixels white. This cleans up the foreground (Figure 8.33) but it will also destroy the edge detail you want to keep. This is where **Clip Rollback** comes in. This is used to put back the edges to restore the detail that was lost. A rather exaggerated clip rollback is shown in Figure 8.34 to illustrate the point.

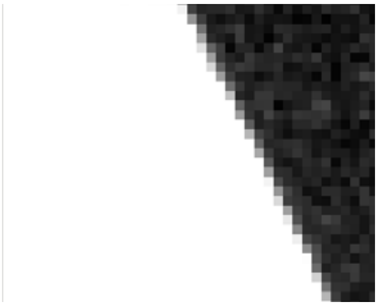


Figure 8.33: **Clip White** has been used to remove the unwanted gray pixels in the white matte.

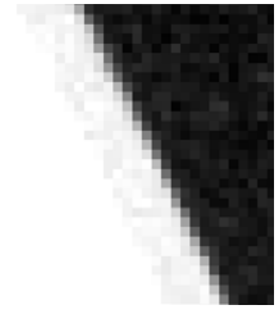


Figure 8.34: **Clip Rollback** has been used to restore the unwanted erosion of the edge.

Dilate

This control should not normally be used as eroding the edges can produce a very poor key. However, the **Screen Dilate** parameter allows you to grow (if greater than zero) or shrink (if less than zero) the alpha in the **Screen Matte**. These controls are sub-pixel accurate.

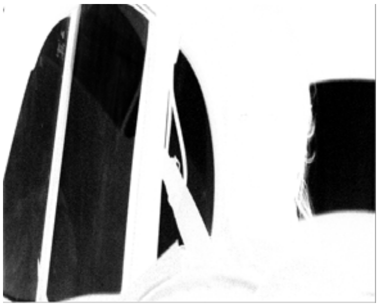


Figure 8.35: Screen Matte.

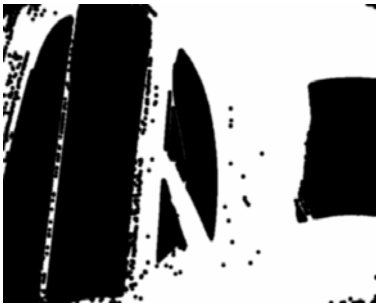


Figure 8.36: Eroded Matte.

Softness

Occasionally, it is useful to be able to blur the matte. Use **Screen Softness** for this. The most common example would be to pull a very harsh matte that you would use as an inside matte further down the tree. For this, you'd soften and erode the screen matte.

Despot

This controls how much to simplify the matte. It coagulates similar regions so that, for example, black specks in the white matte can be absorbed by the surrounding white areas. Increasing the **Screen Despot Black** will remove isolated spots of black in the white matte. Increasing **Screen Despot White** will remove isolated spots of white in the background up to that size.

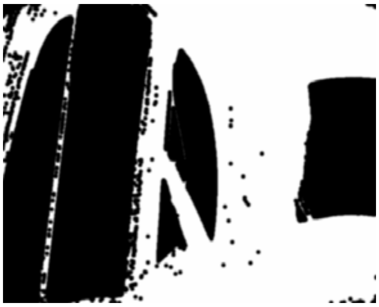


Figure 8.37: Eroded matte.

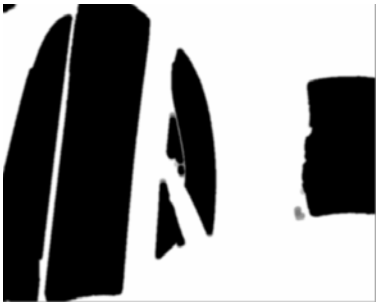


Figure 8.38: Despot.

Mattes

There are 4 mattes in Keylight.

1. Screen Matte
2. Inside Mask
3. Outside Mask
4. Alpha (Composite Alpha)

The **Screen Matte** is generated by the Keylight algorithm after the screen color has been picked. It can be processed (clipped, eroded, etc) by the screen matte processing tools.

The **Inside Mask** is the hold out matte. It is used to confirm areas that are definitely foreground. If your subject has blue eyes and is being shot in front of a blue screen, this mask can be used to put back the eyes. This mask is taken from the **InM** input to Keylight. The embedded alpha channel of the foreground input can be added to this mask using the **Source Alpha** parameter in the **Inside Mask** folder.

The **Outside Mask** is the garbage mask and is used to remove unwanted objects (lighting rigs, etc) from the foreground. The mask is taken from the **OutM** input to Keylight. The luminance or the alpha of this input is set using the **OutM Component** parameter.

The matte used to blend the foreground and background in the final composite is the alpha displayed in the alpha channel of the composite. This matte is the combination of the screen matte, inside matte, and outside matte.

Inside and Outside Masks

If you can't adequately improve the screen matte using the clip levels, you can create a matte in Nuke round the pixels you definitely want to be foreground or background and use this as a mask input. The inside mask makes the foreground less transparent and the outside mask is used to clean up the background that might have bits of the foreground showing through. It is sometimes referred to as the hold out mask.

The outside mask (garbage mask) is often used to clean up screens that are not a constant color or have lighting rigs in shot (Figure 8.39) by forcing the alpha transparent.

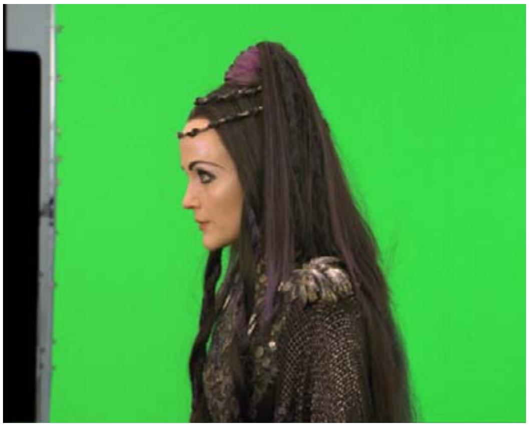


Figure 8.39: Green screen with lighting rig visible.

The inside mask can be used to keep elements in the foreground that you don't want to lose (an actor's blue eyes in front of a blue screen). These masks should normally be softened externally to blend into the screen matte.

Figure 8.40 shows the Bezier spline drawn around the lighting rig on the left side of the screen.

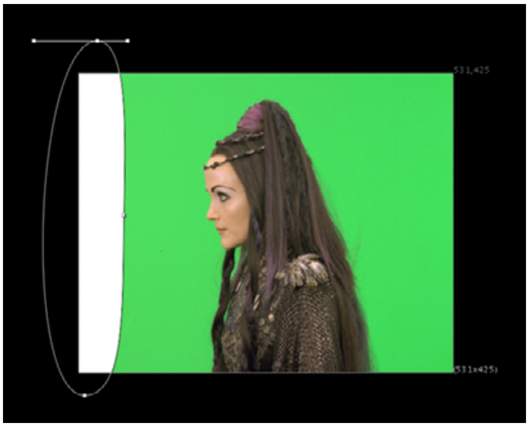


Figure 8.40: Bezier drawn round the lighting rig.

Connect the mask to the **OutM** input of Keylight and switch the parameter **OutM Component** to **Alpha**. The outside mask forces that part of the image to be in the background, thus keying out the rig.

Source Alpha

This parameter determines what to do with any embedded alpha in the original source image. You will need this if you are doing multiple keys on different parts of the image with the View output set to **Intermediate Result**.

- **Ignore** - this will not add any embedded alpha to the screen matte.
- **Add To Inside Mask** - the embedded alpha is added to the inside mask. You should select this when multipass keying with **Output View** set to **Intermediate Result**.
- **Normal** - the embedded alpha is used to composite the image.

Color Replacement

Remember that Keylight does two things - it removes the screen color to despill the image and generates an alpha (**Screen Matte**) to composite the foreground over the background layer.

If you then process the **Screen Matte**, for example, by eroding the alpha or changing the clip levels, Keylight will be removing the wrong amount of screen color from the pixels whose transparency has now changed. The **Screen Replace** instructs Keylight how to deal with such pixels. The **Status** will display which pixels use a replace method. Those pixels that use a replace method because the alpha processing tools modified the transparency will be green, whilst those pixels whose transparency was modified by the inside matte will be blue. See the Status View on page 302.

There are four options to the replace method. These are:

1. **None** - the despilled image is left untouched if the alpha is modified.
2. **Source** - the image will have a corresponding amount of the original pixel (screen color and all) reintroduced/removed if the alpha is changed.
3. **Hard Color** - the despilled image has a corresponding amount of the **Screen Replace Color** added for any increase in alpha.
4. **Soft Color** - the despilled image has a corresponding amount of the **Screen Replace Color** added for any increase in alpha, however, it attempts to modulate the luminance of the resulting pixel so that it matches the original pixel. This will give a more subtle result than the **Hard Color** option.

Inside mask

If the changes to the screen matte are due to an inside mask, the **Inside Replace** and **Inside Replace Color** parameters can be used to modify the color in these areas just like the **Screen Replace** parameters described above.

Edges

Built-in crop tools are included to quickly remove parts of the foreground at the edges of the image. It can also be useful in tidying up a matte at the edges where luminance changes in the blue screen are proving difficult to key out.

With **X Method** and **Y Method** set to **Color** and **Edge Color** set to pure blue (for a blue screen), set the **Left** to crop out the lefthand side of the image revealing the background. Figure 8.41 and Figure 8.42 show the changes to the **Combined Matte** with cropping.

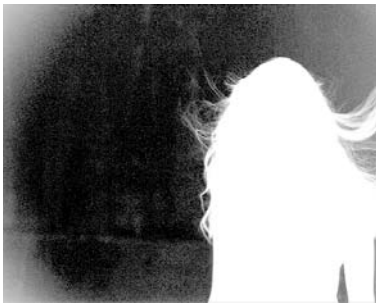


Figure 8.41: **Left** = 0.

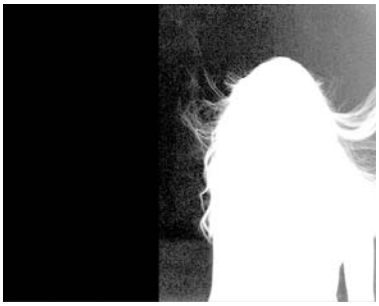


Figure 8.42: **Left** = 0.35.

InM component

The component (luminance or alpha channel) of the inside mask input that is used in the calculations.

OutM component

The component (luminance or alpha channel) of the outside mask. This mask is used as a garbage mask to reveal the background through the foreground as shown in Figure 8.44.

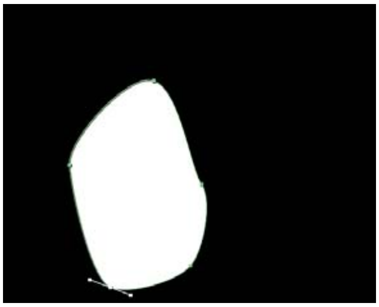


Figure 8.43: Outside Mask.

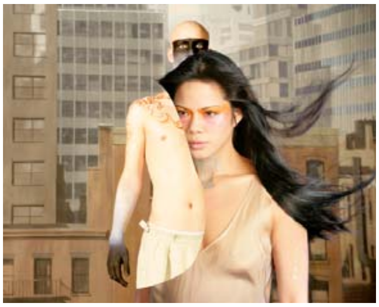


Figure 8.44: Revealing the background.

9 KEYING WITH ULTIMATTE

This section explains how to use the blue/green screen keyer, Ultimatte, in Nuke.

Ultimatte Quick Start

To get you started swiftly with using Ultimatte, here's a rough overview of the workflow:

1. Sample the screen (backing) color. For more information, see "Sampling the Screen Color" on page 320.
2. Refine the overlay using the **overlay** tools and, if needed, adjust the controls on the **Ultimatte** tab. For more information, see "Using Overlay Tools and Screen Correct" on page 321.
3. Next refine the matte density using the **matte** tools and, if needed, adjust the controls on the **Density** tab. For more information, see "Adjusting the Density of the Matte" on page 323.
4. If necessary, activate Shadow processing, and use Shadow tool and, if needed, adjust the controls on the **Shadow** tab. For more information, see "Retaining Shadows and Removing Noise" on page 325.
5. If necessary, improve Spill Suppression using **spill** tools and, if needed, adjust the controls on the **Spill** tab. For more information, see "Adjusting Spill Controls" on page 324.
6. If necessary, adjust the **Cleanup** controls, but in general you'll get better results by using **Screen Correct** and **Matte Density** controls. For more infromation, see "Retaining Shadows and Removing Noise" on page 325.
7. If necessary, you can adjust the controls on the Color tab to match your blacks, whites and gammas between the foreground and the background. For more information, see "Adjusting Color Controls" on page 326.
8. If necessary, activate film processing and adjust its settings on the Film tab. For more information, see "Adjusting Film Controls" on page 327.

Connecting the Ultimatte Node

Start up Nuke, create an Ultimatte node by clicking **Keyer** > **Ultimatte** and connect a foreground and a background image (and any additional inputs you want) to it. Add a Nuke Viewer node so you can see the result.

To use the Ultimatte inputs

In Ultimatte, you have a number of inputs you can use to connect images and mattes that you need to key your footage. Connect the following inputs to your images and/or mattes as necessary:

- **foreground (fg)** - Connect your foreground image to this input.
- **background (bg)** - Connect your background image to this input.
- **clean plate (cp)** - Connect your clean plate to this input. For more information on clean plates, see "Using Overlay Tools and Screen Correct" on page 321.
- **garbage matte (gm)** - Connect your garbage matte to this input. A garbage matte is often used to clean up screens that are not a constant color or have lighting rigs in shot by forcing the alpha transparent.
- **holdout matte (hm)** - Connect your holdout matte to this input. A holdout matte is used when foreground objects are the same color or very close to the backing colora. The holdout matte is used to let Ultimatte know that the pixels in the holdout matte region should be considered 100% opaque. Note that, unlike the other inputs, this input is hidden and appears as a small triangle on the left hand side of the node.

Sampling the Screen Color

The first step when keying with Ultimatte is to sample the screen color, or tell Ultimatte what color your blue or green screen is. Do the following:

1. When you've connected the Ultimatte node, the foreground image is displayed in the Viewer.
2. Click **Screen** in the Ultimatte toolbar above the Viewer, and select the screen color by holding down **Alt+Ctrl/Cmd** and clicking in the Viewer.

   You should choose an area on the green screen near important subject detail that is not obscured in any way. The image is rendered and a composite is displayed. If further adjustments are needed, use the controls described below.

Adjusting the controls on Ultimatte tab

After you've sampled your screen color, you can start adjusting your result. With the controls on the **Ultimatte** tab you can choose different sets of controls you want to enable on other control panel tabs. There's also an **enable** checkbox on each tab that you can use to activate the controls on that tab and the corresponding tools in the Viewer.

- **Film** - check this to to reduce the effects of cyan undercutting on the **Film** tab.
- **screen correct** - check this to use the controls on **Screen Correct** tab.
- **shadow** - check this to use the controls on the **Shadow** tab.

- **spill suppression** - check this to use the controls on the **Spill** tab.
- **cleanup** - check this to use the controls on the **Cleanup** tab.
- **color conformance** - check this to use the controls on the **Color** tab.

Using Overlay Tools and Screen Correct

Screen Correct compensates for anomalies in the backing area such as uneven lighting, smudges, seams, variation in the backing color, green set pieces, and unwanted shadows cast by set pieces, boom arms, etc. This technique assumes that an exact copy of the problematic green screen element with the subject matter omitted, usually called a Clean Plate, is supplied.

Although this technique gives great results by differentiating between foreground elements and backing flaws, you often haven't got the necessary reference materials. In that case, you can create a synthetic Clean Plate with Ultimatte.

To achieve the best results, use a reference Clean Plate as an input to Ultimatte in addition to allowing Ultimatte to generate a synthetic plate. In this way, the reference plate allows for the best shadow separation, while the synthetic plate is be used to reduce such artifacts as film grain or video noise (which is rarely accurately reproduced when the clean plate is shot during the time of principal photography). Switch the view to **screen** by clicking the **overlay** dropdown and selecting **screen**.

With **screen correct** selected, use the **add overlay** tool to scrub on areas that represent the green screen, including shadows. Usually, it's best to press **Alt+Ctrl/Cmd** while scrubbing, so that you are picking the color in the input image at that pixel, instead of the color of the processed image.

The overlay will be used in the final process to fully eliminate these areas from the final composite. Continue in this manner until the foreground subject and its shadows are surrounded with an overlay. Make sure the overlay does not cover any foreground detail that is to be retained. If the overlay does cover foreground subject matter, then use the **remove overlay** tool to scrub on those areas that should not be covered by the overlay. Repeat these two processes until the overlay covers the green screen areas and no subject matter. View the composite in the Viewer to see the screen corrected composite with anomalies in the backing area removed. To learn which controls were automatically set by Ultimatte, click the **Screen Correct** tab and note the position of the controls.

Note *When scrubbing on the image using **add overlay**, the RGB value of the points chosen are accumulated in a keep list and **remove overlay** points are accumulated in a drop list. If both lists have equivalent values, a conflict may occur, resulting in no visible change in the overlay. If a conflict occurs, try using Undo (**Ctrl/Cmd+X**) which flushes the last set of points added to the appropriate list. If that doesn't help, you can also press the **Reset** button on the **Screen Correct** tab to clear the lists and start over.*

Under some circumstances, it may be difficult to completely cover the screen area with the overlay. This does not mean that the screen will not be removed in that area, but that other controls may need to be adjusted to help reduce anomalies in those areas that were not covered by the overlay. It may be advantageous to resample the screen using the **Screen** tool above the Viewer in the area where covering the screen with the overlay is unsuccessful, but be careful about selecting in shadow areas if the final composite requires the retention of shadows.

Additionally, it may be difficult in some circumstances to remove the overlay from very dark areas of foreground detail that may be close to values in dark or shadow areas of the screen. If the overlay does include some dark foreground areas, these areas may be recovered by enabling the **Shadows** controls.

Note *Since Ultimatte produces a synthetic clean plate using an interpolation process, it is important to exclude non-video areas of the image, such as the black blanking lines that may appear in video or the black areas in letterbox images. You can use other nodes, such as Crop, to exclude these areas from the overlay and final composites. Otherwise the synthetic clean plate may contain black (or other colors) which can result in an inaccurate clean plate being generated.*

Adjusting overlay controls

You can choose your overlay view, color and output mode in the control panel, on the **Ultimatte** tab.

- **overlay** - The **overlay** control is used to show the calculated overlay on the Viewer and it's helpful for debugging purposes. It helps tune the **screen correct** controls and also the **add/remove overlay** tools. In this way you can see immediately if something is wrong and needs to be adjusted. In the overlay drop-down menu, change
  - **off** - to not view the overlay.
  - **screen** - to view the subject in clear, and the preliminary matte area blended with the overlay color.

  - **subject** - show the subject blended with overlay color, and the preliminary matte area in clear.
- **show image as monochrome** - check this to make the input image appear in greyscale so that the overlay areas can be more easily distinguished.
- **overlay color** - use this to change the color of the overlay. You can adjust the alpha channel to modify the opacity of the overlay.

Adjusting the screen correct controls

- **screen tolerance** - This is used to adjust the color range or tolerance to be included or excluded from the screen overlay.
- **shrink** - Use this control to shrink the screen overlay.
- **darks(reds smaller)** - Use this control to include or exclude dark areas from the screen overlay in those areas where the blue value (when using green screen)is greater than the red value in the original foreground image.
- **darks(reds larger)** - Use this control to include or exclude dark areas from the screen overlay in those areas where the red value is greater than the blue value (when using green screen) in the original foreground image.
- **brights(reds smaller)** - Use this control to include or exclude bright areas from the screen overlay in those areas where the blue value (when using green screen) is greater than the red value in the original foreground image.
- **brights(reds larger)** - Use this control to include or exclude bright areas from the screen overlay in those areas where the red value is greater than the blue value (when using green screen) in the original foreground image.
- **orphans** - Use this to include or exclude neighboring "rogue" pixels from the screen overlay.

Adjusting the Density of the Matte

The controls on the **Density** tab are used to adjust the density or opacity of the foreground objects. The density of a foreground object is determined by its matte (alpha) value. A completely opaque object's matte will be white, a completely transparent object's matte will be black, and a partially transparent object's matte will be gray. Use the **add matte** dropper to scrub on areas in the matte that appear gray, but should be white (fully opaque) in the matte. These areas are described as "print-through", meaning that the opacity of the subject is too low in this area and the background is visible through the foreground in this area. Be careful not to select those objects that should have a gray matte value such as fine hair, smoke or partially transparent objects, as they will become opaque. To learn which controls were automatically set by Ultimatte, click the **Density** tab in the control panel and note the position of the controls.

Note *If there are sections of the matte which should be opaque but are exhibiting gray values that don't look like typical transparency, then there is a chance that there is a remainder of overlay in this area. If overlay exists on subject matter, return the **Overlay** drop-down menu to **Screen** and use the **remove overlay** tool to scrub on that area. Check on **show image as monochrome** on the **Ultimatte** tab to aid in determining the extent of the overlay.*

Adjusting Density controls

Use these controls to adjust the density of your matte:

- **brights** - Use this control to adjust density in bright foreground objects. Advancing this control too far can cause hard, dark edges around foreground subjects.
- **darks** - Use this control to adjust density in black glossy or dark foreground objects.
- **edge kernel** - Use this control to adjust number of pixels to use as a kernel to reduce dark edges that may exist in transition areas due to an over-dense matte. Advancing this control too far, may cause too much print-through at the edges.
- **warm** - Use this control to adjust density in warm colors (flesh tones). Note that reducing this control too much can cause print-through in reddish foreground objects.
- **cool** - Use this control to adjust density in cool colors. Note that reducing this control too much can cause print-through in bluish foreground objects.

Adjusting Spill Controls

Ultimatte automatically suppresses spill from the backing onto foreground subject matter. The spill controls are used to suppress excessive spill or to retain color similar to spill that has been over-suppressed. To adjust the spill controls, check the **spill suppression** box on the **Ultimatte** tab and click **Spill** tab:

- **cool** - Use this control to adjust the amount of spill in cool colored foreground objects. Used to reproduce blue, green or cyan colors that changed through the spill suppression algorithms.
- **warm** - Use this control to adjust the amount of spill in warm colored foreground objects. Used to reproduce pink, purple and magenta colors for bluescreen, or yellow and orange colors for green screen that changed through the spill suppression algorithms.

- **midtones** - Use this control to adjust the amount of spill in midrange foreground objects.
- **brights** - Use this control to adjust the amount of spill on bright foreground objects.
- **darks** - Use this control to adjust the amount of spill on dark foreground objects.
- **ambience** - Use this control to choose a color to subtly influence the foreground objects in areas that may have contained spill.
- **strength** - Use this control to adjust the intensity of the chosen **ambience** color.
- **background veiling** - This control is used to override the automatic suppression of the backing color. Ultimatte uses the selected backing color to suppress the backing to black. An indication that the automatic settings did not suppress enough backing is "veiling" or a colorized haze in some background areas. An indication that the automatic settings suppressed too much backing is darkened or mis-colored foreground edges and transparencies. In most cases this control should be left at the default setting.

Retaining Shadows and Removing Noise

Use the **hold shadow** dropper (only available when **screen correct** and **shadow** are enabled) to scrub on the shadows that you'd like to preserve. These shadows may best be seen in the foreground image.

If unwanted shadows remain, then use the **remove noise** dropper to reduce or eliminate those shadows. If the area that you scrub does not reside under the overlay, then erasing adjusts the appropriate **Cleanup** controls to reduce anomalies in the screen area. This might result in losing some fine detail. Repeat using these two tools until the shadows you want are retained and any shadows or noise you don't want to keep are removed.

The density, sharpness, and tint of the shadow may be altered by manually adjusting the shadow controls in the **Shadow** controls. To learn which controls were automatically set by Ultimatte, click the **Shadow** and/or **Cleanup** tab in the control panel and note the position of the controls.

Adjusting Shadows controls

Check the **shadow** box on the **Ultimatte** tab and adjust the following controls:

- **high** - Decrease this to reduce or eliminate unwanted shadows (which must be lighter than those shadows that are to be retained). All preserved shadows will be lighter.

- **low** - Increase this to restore the density of the darkest part of the shadows that are to be retained.
- **density** - Use this to change the density of the shadows that are retained to better match shadows in the background scene.
- **blur** - Use this to blur the shadows.
- **tint** - Use this to tint the shadows.

Adjusting Cleanup controls

The following controls are used to adjust the black and gray areas of the matte channel. This will dramatically affect the nature of foreground objects' edges, the opacity of transparent objects, and the noise in the foreground image. Use these controls sparingly as they might result in the loss of foreground detail. Using **Screen Correction** for dealing with imperfections in the screen is a good alternative for using **Cleanup**.

- **cleanup** - Use this control to reduce imperfections or small amounts of noise in the backing. Note that adjusting this control too far will result in a "cut and paste" look with a hard, unnatural edge. Background noise (as well as some foreground detail) will be reduced. An alternative method for dealing with green screen imperfections is to use the **Screen correct** controls.
- **shrink** - Use this control to choke or reduce the size of the cleaned-up matte.
- **blur** - Use this control to soften the cleaned-up matte.
- **recover** - Use this control to set a threshold below which the **Cleanup** control will not have influence.

Adjusting Color Controls

Check the **color conformance** box on the **Ultimatte** tab, and set these controls on the **Color** tab. A good keying result requires matched blacks, whites, and gammas between the foreground and background elements. With color conformance you can select blacks, whites, and gammas and can automatically match the foreground to the background (or vice versa).

- **darks** - You can use this control to influence the darkest parts of the image. This is a global control that will affect the entire image, but the greatest effect will be seen in the darkest areas.
- **midtones** - You can use this control to influence the mid-range parts of the image. This is a global control that will affect the entire image, but the greatest effect will be seen in the mid-range areas.
- **brights** - You can use this control to influence the brightest parts of the image. This is a global control that affects the entire image, but you can see the greatest effect in the brightest areas.

- **hue** - This control changes the color contents of the image without changing its brightness or color intensity (purity) values. At default setting (0), the image hue is not altered. The range of the control extends from -300 to +300.
- **saturation** - This control changes the color intensity or purity values of the image without altering its color contents or brightness values. At default setting (0), the image saturation is not altered. At minimum setting (-200), the color intensity is reduced to zero and the image is monochrome, or shades of gray.
- **brightness** - This control changes the overall intensity of the image. At the default setting, the image brightness is not altered.

Adjusting Film Controls

Use the controls on the **Film** tab to reduce the effects of cyan undercutting. These controls are only available when **Film** box is checked on **Ultimatte** tab. You can view the results better when you select **subject** in the **overlay** drop-down on the **Ultimatte** tab.

Due to the nature of film's emulsion layers, a phenomenon know as cyan undercutting exists that reveals itself as a smeared red edge in areas of sharp transitions, which can best be seen by viewing the individual RGB channels of a film image. Normally, this phenomenon is not a problem until bluescreen compositing techniques are applied. Since the red emulsion layer tracks at a slower rate than the blue and green emulsion layers, the artificial red edge it produces are retained as foreground detail resulting in an unacceptable composite.

- **transparency** - Use this control to adjust the amount of Film Correction in partially transparent foreground objects (such as hair detail).
- **correction** - Use this control to adjust the amount of correction in the individual RGB channels of the foreground image.
- **strength** - Use this control to adjust the overall amount of Film Correction that is applied to the foreground image.
- **shrink** - Use this control to shrink the subject overlay.
- **brights** - Use this control to include or eliminate bright areas from the subject overlay.
- **darks** - Use this control to include or eliminate dark areas from the subject overlay.

Choosing an Output Mode

You can output the merged foreground and background as a final composite, or you can output the premultiplied or unpremultiplied foreground for compositing elsewhere in your node tree. In the **output mode** dropdown, choose:

- **composite** - to output a merged layer of the background input with the extracted subject.
- **premultiplied** - to output the extracted subject premultiplied by the final matte.
- **unpremultiplied** - to output the extracted subject and final matte unpremultiplied.

10 USING ROTOPAINT

Nuke features a vector-based RotoPaint node for help with tasks like rotoscoping, rig removal, garbage matting and dustbusting. You can draw Bezier and B-Spline shapes with individual and layer group attributes, including per-point and global feather, motion blur, blending modes and individual or hierarchical 2D transformations. This chapter gives full instructions on its usage.

Roto or RotoPaint?

There are two similar nodes in Nuke for doing rotoscoping with, Roto and RotoPaint. The main difference between these two is that you can only create and edit Bezier and B-spline shapes with Roto, while RotoPaint allows you to draw paint strokes too with various brushes. So the Roto node is an optimal choice if you're doing rotoscoping only, whereas RotoPaint gives you a broader scale of tools to use.

All tools and controls in the Roto node work the same way as they do in RotoPaint, so you can learn about using them in the RotoPaint instructions in this chapter. For instance, see:

- "Working with the Toolbars" on page 331 for information about the toolbars in Roto
- "Working with the Stroke/Shape List" on page 331 for information about the shape list in the Roto control panel.
- "Drawing Shapes" on page 343 for information about using the Bezier and B-Spline Tools.
- "Selecting Existing Strokes/Shapes for Editing" on page 351 for information about selecting shapes.
- "Editing Shape Specific Attributes" on page 358 for information about editing Bezier and B-Spline attributes.
- "Editing Existing Stroke/Shape Timing" on page 362 for information about changing the timing of your shape.
- "Animating Strokes/Shapes" on page 365 for information about editing your shapes.

RotoPaint Quick Start

To get you started swiftly with using RotoPaint, here's a rough overview of the workflow:

1. Connect the RotoPaint node to a Viewer and possible backgrounds. For more information, see "Connecting the RotoPaint Node" on page 330.

2. Select a stroke or a shape tool from the RotoPaint toolbar on the left side of the Viewer. For more information, see "Drawing Paint Strokes" on page 333 or "Drawing Shapes" on page 343.
3. Use the RotoPaint tool settings to adjust the stroke/shape you're about to draw. For more information, see "Working with the Toolbars" on page 331.
4. Draw one or more strokes/shapes in the Viewer window. For more information, see for example "Using the Brush tool" on page 335 or "Using the Bezier Tool" on page 344.
5. Select an existing stroke/shape using the Select tools or the stroke/shape list. For more information, see "Selecting Existing Strokes/Shapes for Editing" on page 351.
6. Use the contol panel controls to adjust your existing stroke/shape. For more information, see "Editing Existing Stroke/Shape Attributes" on page 353.

In addition you can:

- Adjust the splines of your stroke/shape. For more information, see "Editing Existing Stroke/Shape splines" on page 363.
- Animate your strokes/shapes. For more information, see "Animating Strokes/Shapes" on page 365
- Use RotoPaint in stereoscopic projects. For more information, see "RotoPaint and Stereoscopic Projects" on page 370.
- Set your favorite RotoPaint tool as your default tool. For more information, see "Setting Default RotoPaint Tools and Settings" on page 348.

Connecting the RotoPaint Node

The RotoPaint node accepts one primary input. Even if you have no inputs, you can still use RotoPaint and in that case, you can use the **format** control to choose your output format.

To connect the RotoPaint node:

1. Click **Draw > RotoPaint** to add a new RotoPaint node. You can also press **P** on the Node Graph. To create a Roto node, you can press **O** on the Node Graph.
2. Drag the **bg** input to the node that you want to apply RotoPaint to.
3. Connect any additional background elements you wish to use. If you plan to reveal pixels from a background element, drag the **bg1** input from the left side of the node to the node whose output you wish to use.

Working with the Toolbars

In the RotoPaint node, you can use two toolbars to define the type of stroke/shape you want to start drawing. These toolbars are placed in the Viewer. The vertical RotoPaint toolbar is for selecting the tool you want to use and the horizontal one, RotoPaint tool settings, is for adjusting the currently selected tool's settings before drawing new strokes/shapes.

Note *You can't use RotoPaint tool settings to adjust an already existing stroke/shape. For any changes you want to make to a stroke/shape after you've created one, you can use the controls in the RotoPaint control panel.*

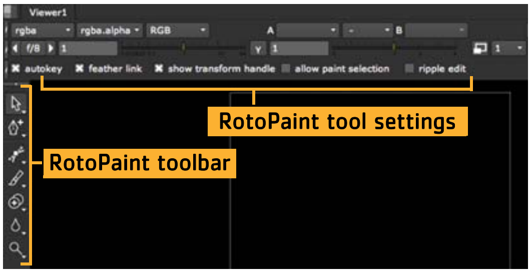


Figure 10.1: RotoPaint toolbar and tool settings.

In the RotoPaint toolbar, you can select your stroke/shape tool. The tools are grouped under the toolbar icons. You can click any tool to make it active and view a tool group by right-clicking (or left-clicking and holding) the icon. The tool that is currently selected is highlighted.

In the RotoPaint tool settings on the top of the Viewer, you can define the settings for the tool that you've chosen. The controls in this toolbar change depending on which tool you have selected at any given time.

Tip *You can also hide the toolbars by clicking the hide button next to them. You can also press [(square bracket) to hide the RotoPaint toolbar and { (curly bracket) to hide the tool settings (and the entire top toolbar of the Viewer).*

Working with the Stroke/Shape List

After you've drawn strokes/shapes, you can edit their order and group them with the Stroke/Shape list in the RotoPaint control panel. By default, the newest stroke/shape/group will appear on top of the list, and your strokes/shapes are named according to their type (for example, "Bezier1" or "Smear2").

Using the list, you can also select strokes/shapes/groups and edit them with the various controls in the control panel. Some controls can be applied to groups, some can't. Some controls also can only be applied to strokes or shapes (these are grouped under the **Stroke** and **Shape** control tabs respectively). If a control can't be applied to a group, it will be grayed out if you have a group selected in the stroke/shape list.

The stroke/shape list provides an overview of existing parameters, showing, for example, whether the item is locked, set invisible, or whether it has motionblur applied to it. Some controls can also be edited directly in the overview by clicking on their icon.

To use the stroke/shape list:

You can edit the stroke/shape list in many ways, and use it to adjust strokes/shapes and how they're displayed in the Viewer.

- You can reorder the columns in the stroke/shape list by dragging and dropping them.
- You can create groups for sets of strokes/shapes in the stroke/shape list by clicking the **Add** icon below the list. This will create a subfolder, named "Layer1" by default, and you can drag and drop strokes/shapes to this folder to group them. After strokes/shapes have been grouped, you can edit them as a group and they will move together if you change their place in the Viewer. Every group also has its own transformation overlay, which you can use to move the group.
- You can remove a stroke/shape or a group by clicking the **Remove** button under the stroke/shape list.
- If you want to rename any of the strokes, shapes, or groups, double-click on the name while the item is selected, and give your item a new name. The name has to be unique though, so you can't give two items the same name.
- You can also cut, copy, and paste strokes and shapes by right-clicking on them and using the **copy**, **cut**, and **paste** options in the menu that appears. If you paste a stroke/shape on top of another stroke/shape, the position and attributes of the original stroke/shape will be replaced with the pasted stroke/shape.
- You can duplicate strokes/shapes by right-clicking on them and selecting **duplicate**. A new stroke/shape is created with the same spline, animation and attributes as the one you selected.
- You can hide a stroke, shape, or group by clicking the **Visible** icon in the stroke/shape list. You can still edit an invisible stroke/shape and view its position in the Viewer.
- You can lock strokes/shapes to prevent them from being edited. To lock an item in the stroke/shape list, click the **Lock** column in the list. A lock icon appears next to the **Visible** icon.

- You can choose the color in which you want the outline of your stroke/shape to appear in the Viewer. Click the **Overlay** column and choose your overlay color. To be able to see the overlay in the Viewer, you need to activate one of the Select tools in the RotoPaint toolbar and check **allow paint selection** in the tool settings.
- You can change the color of your stroke/shape in the stroke/shape list by clicking the **Color** column and use the color picker to choose the color.
- You can invert a shape using the **Invert** column. With your shape selected, click in the **Invert** column to toggle between inverted and uninverted modes.
- You can choose a blending mode for your stroke/shape using the **Blending** column. With your shape selected, click the Blending column and choose the mode you want to apply.
- You can apply motion blur to your shape using the **Motion blur** column. With your shape selected, click the **Motion blur** column to toggle the motion blur effect.

Tip *To undo and redo any changes you've made with the RotoPaint node, use the **Undo** and **Redo** buttons on top of the control panel. Undo uses a cache memory of your changes, so at any time you can undo and redo all the changes you've made since you last opened your project.*

Drawing Paint Strokes

Any given RotoPaint node can hold many paint strokes and shapes. You can apply paint strokes using any of the following tools (see also "Drawing Shapes" on page 343).

Icon	Tool	Hotkey	Function
	Brush	**N** (toggles between Brush and Eraser)	Applies colors atop the current plate, or blends colors with the current plate. You can also clone from surrounding frames.
	Eraser	**N**	Removes pixels from existing strokes and brings background back.
	Clone	**C** (toggles between clone and Reveal)	Applies pixels from one region of the current plate to another region of the current plate.
	Reveal	**C**	Applies pixels from a source plate to a destination plate in the corresponding place.
	Blur	**X** (toggles between Blur, Sharpen and Smear)	Blurs the image in the area of the brush stroke.

Icon	Tool	Hotkey	Function
Δ	Sharpen	**X**	Sharpens the image in the area of the brush stroke.
	Smear	**X**	Smears the area of the smear brush stroke, stretching the selected pixels over their surrounding area.
	Dodge	**D** (toggles between Dodge and Burn)	Brightens the background color on the area of the brush stroke to reflect the brush stroke. Using this tool on black produces no change. No part of the stroke area will be darkened.
	Burn	**D**	Darkens the background color on the area of the brush stroke to reflect the brush stroke. No part of the stroke area will get lighter.

Tip *You can choose to have your RotoPaint always open with a particular tool selected. If you want to open it with the Brush tool selected, for example, do the following:*

1. Create a file called menu.py in your plug-in path directory if one doesn't already exist. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. To select Brush tool as your default tool, save the following in your menu.py:

nuke.knobDefault('RotoPaint.toolbox','brush')

3. Restart Nuke.

For more information on default tools, see "Setting Default RotoPaint Tools and Settings" on page 348.

Tip *If you are using a tablet, you can tie a new stroke's opacity, size, or hardness to pen pressure by checking the corresponding box in the RotoPaint tool settings. You can also later change which attribute the pressure alters for an existing stroke in the control panel.*

Separate Select tools in the RotoPaint toolbar let you select strokes/shapes once they've been drawn and after that you can make changes to them using the control panel controls (see "Selecting Existing Strokes/Shapes for Editing" on page 351, and "Editing Existing Stroke/Shape Attributes" on page 353).

This section discusses the general steps for using each of these tools, and gives an overview on editing their attributes and timing.

Using the Brush tool

The **Brush** tool lets you apply colored or blended strokes to the current plate.

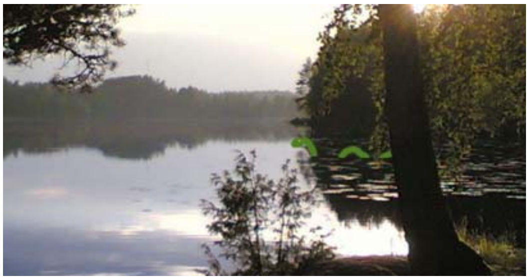


Figure 10.2: Painting with the Brush tool.

To use the Brush tool

1. Click the **Brush** tool in the RotoPaint toolbar.
2. Set color, opacity, brush type, brush size, and brush hardness in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.) You can also change the size of the brush using **Shift**+drag directly in the Viewer as shortcut.
3. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
4. Apply strokes as necessary.

Using the Eraser Tool

The **Eraser** tool lets you remove pixels from existing paint strokes.

To use the Eraser tool

1. Set opacity, brush type, brush size, and brush hardness in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Selecting Existing Strokes/Shapes for Editing" on page 351.)
2. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)

3. Right-click the **Brush** tool in the RotoPaint toolbar and select the **Eraser** tool.

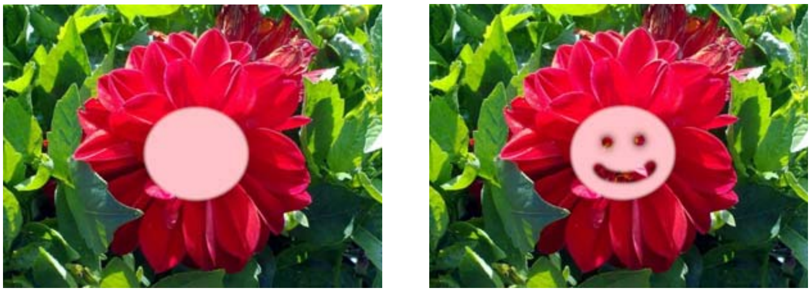


Figure 10.3: Painting with the Eraser tool.

4. Apply strokes as necessary. You can also erase multiple strokes, if you have drawn more than one.

Tip *If you're using a graphics tablet, Nuke automatically switches to Eraser mode when the erase end of the pen is used.*

Using the Clone Tool

The **Clone** tool lets you remove unwanted features from the plate or from a different input by painting over them with pixels offset from the pointer or a transformation of the pointer.

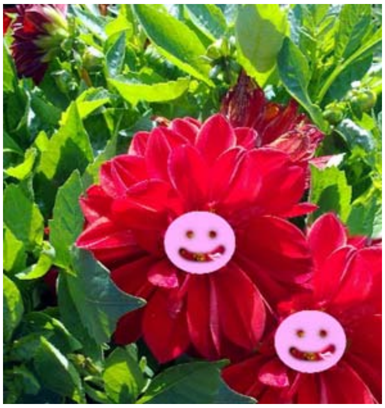


Figure 10.4: Painting with the Clone tool.

To use the Clone tool

1. Click the **Clone** tool in the RotoPaint toolbar. To view all the settings for the Clone tool, click the **Show Clone Settings** button on the RotoPaint tool settings.

2. In the RotoPaint tool settings at the top of the Viewer, set the paint source menu to the input you want to clone pixels from. (For information on the available options, see "Selecting a source image" on page 359.)

   You can also use the transform controls in the clone settings to transform the clone source and reset it back to original using the **reset transform** button.
3. Set opacity, lifetime, brush type, size, and hardness for your stroke in the RotoPaint tool settings. (For more information, see "Editing Existing Stroke/Shape Attributes" on page 353 and "Editing Existing Stroke/ Shape Timing" on page 362.) If you've got more than one view set up, you can check the **single view** box to only clone in one view, or uncheck it to clone in all of them.
4. To set the offset, hold down **Ctrl/Cmd** and left-click at the source location, drag to where you want to paint, and release. Alternatively, you can enter the offset numerically using the **translate** controls in the RotoPaint tool settings. If you'd like the offset amount to be rounded to an integer (whole number of pixels), check **round**. Rounding to a pixel can be useful if you don't want to soften the image by partially blending pixels together.
5. Start painting. The pointer overlay depicts the source of the offset as a crosshair within a circle and the destination as a circle (the diameter of which represents the breadth of the stroke).

   You can use **/** (forward slash) and ***** (asterisk) on the numeric keypad to zoom your clone source in and out, and **0** (zero) and **.** (decimal point) to rotate it right and left. You can also use the number keys on the numeric keypad to nudge the clone source.
6. To reset the cloning offset, you can use **Ctrl/Cmd**+drag to adjust the offset you set before, or **Ctrl/Cmd+Shift**+drag to start a new offset from the brush pointer's location.

Tip *If you're cloning from the current plate (**foreground**), you're also cloning all the strokes/shapes you've previously drawn. If you want to clone from the original background or a different picture, you need to set the paint source menu to pull from that input.*

Tip *To clone pixels from another frame of the input clip, you can use **time offset** slider to define which frame you want to clone from. See "Editing Clone or Reveal Attributes" on page 362.*

Using the Reveal Tool

The **Reveal** tool lets you pull pixels from background elements onto the current plate. The **Reveal** tool requires at least two inputs (see "Connecting the RotoPaint Node" on page 330); otherwise, your strokes will draw in white. You can also view which input you are using as the source for your strokes in the **Source** column in the stroke/shape list.

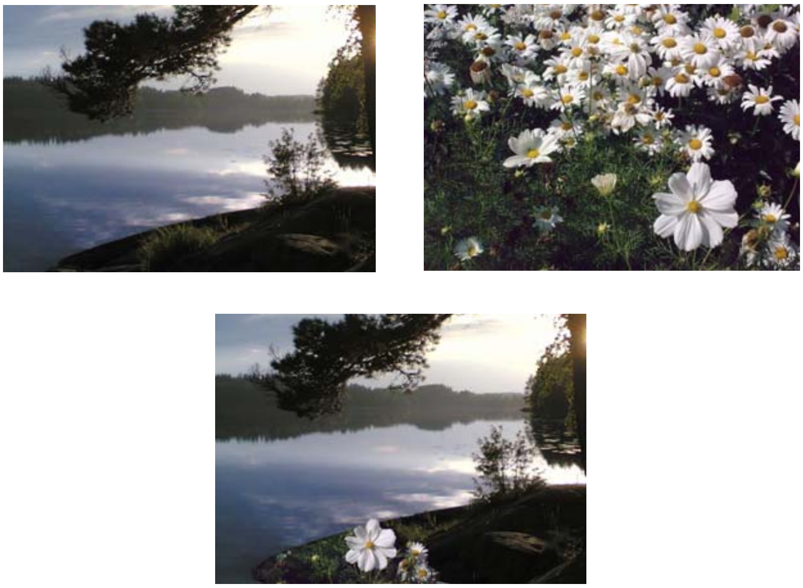


Figure 10.5: Painting with the Reveal tool.

To use the Reveal tool:

1. Right-click the **Clone** tool in the RotoPaint toolbar and select **Reveal** tool.
2. In the RotoPaint tool settings at the top of the Viewer, set the paint **source** menu to the input you want to pull pixels from. (For information on the available options, see "Selecting a source image" on page 359.)
3. Set opacity, brush size, and brush hardness in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
4. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
5. You can also reveal pixels from another frame of the input clip by using the **time offset** slider to define which frame you want to reveal from. See "Editing Clone or Reveal Attributes" on page 362.
6. If you want, you can view your revealing source image in a Viewer overlay. To do this, check the **onion** box in the RotoPaint tool settings and enter an onion skin value to adjust the opacity of the overlay. This

can help you better see what you are revealing from the source image. You can also toggle the hotkey **T** to enable or disable onion skin.

7. Start painting. The pointer overlay depicts both the source of the offset and the destination as a circle (the diameter of which represents the breadth of the stroke).

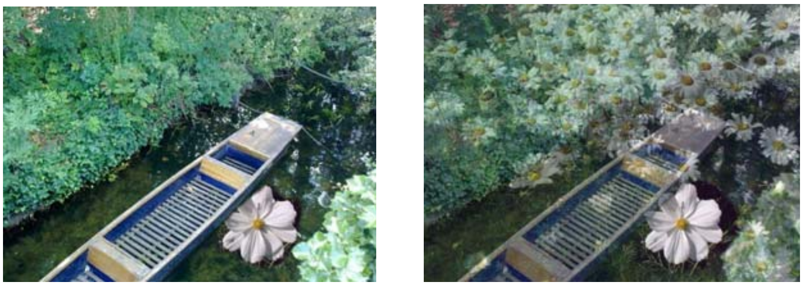


Figure 10.6: Cloning a flower with onion skin disabled (left) and enabled (right).

Using the Blur Tool

The **Blur** tool lets you blur parts of the plate.

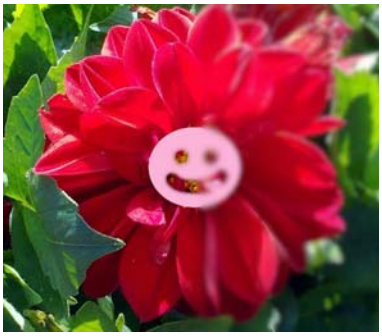


Figure 10.7: Painting with the Blur tool.

To use the Blur tool:

1. Click the **Blur** tool in the RotoPaint toolbar.
2. Set opacity, brush type, brush size, and brush hardness in the RotoPaint tool settings. (For information on the available options, see “Editing Existing Stroke/Shape Attributes” on page 353.)
3. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see “Editing Existing Stroke/Shape Timing” on page 362.)
4. Apply strokes by clicking on the part of image you want to blur.

Tip *You can use the **effect** control in the RotoPaint tool settings to adjust the strength of the tool you're using. With the **Blur** tool, it controls the amount of blur and with the **Sharpen** tool it controls how much the image will be sharpened.*

Using the Sharpen Tool

With the **Sharpen** tool, you can sharpen the image within the area of the brush stroke.

Figure 10.8: Painting with the Sharpen tool.

To use the Sharpen tool:

1. Right-click the **Blur** tool and select **Sharpen** tool.
2. Set opacity, brush type, brush size, and brush hardness in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
3. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/ Shape Timing" on page 362.)
4. Apply strokes by clicking on the part of image you want to sharpen.

Tip *You can use the **effect** control in the RotoPaint tool settings to adjust the strength of the tool you're using. With the **Blur** tool, it controls the amount of blur and with the **Sharpen** tool it controls how much the image will be sharpened.*

Using the Smear Tool

With the **Smear** tool, you can smear or stretch pixels over the surrounding pixels.

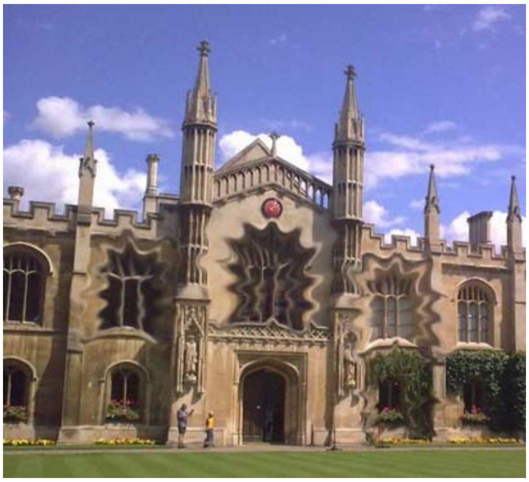


Figure 10.9: Painting with the Smear tool.

To use the Smear tool

1. Right-click the **Blur** tool and select **Smear** tool.
2. Set opacity, brush type, brush size, and brush hardness in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
3. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
4. Apply strokes by clicking and dragging on the part of image you want to smear.

Using the Dodge Tool

With the **Dodge** tool, you can lighten the pixels in the area of the brush stroke. This makes the background color brighter on the area of the brush stroke to reflect the brush stroke.

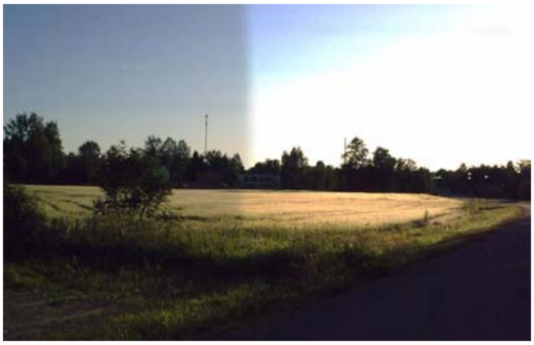


Figure 10.10: Painting with the Dodge tool.

To use the Dodge tool

1. Click the **Dodge** tool.
2. Set opacity, brush type, brush size, and brush hardness in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
3. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
4. Apply strokes as necessary.

Using the Burn Tool

With the **Burn** tool, you can darken the pixels in the area of the brush stroke. This makes the background color darker on the area of the brush stroke.

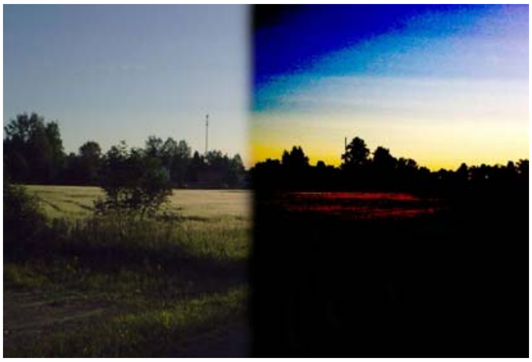


Figure 10.11: Painting with the Burn tool.

To use the Burn tool

1. Right-click the **Dodge** tool and select **Burn** tool.
2. Set opacity, brush type, brush size, and brush hardness in the RotoPaint tool settings. (For more information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
3. Optionally, set the lifetime of the stroke in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
4. Apply strokes as necessary.

Drawing Shapes

Any given RotoPaint node can hold several shapes, and you can draw them using any of the following tools.

Icon	Tool	Hotkey	Function
	Bezier	**V** (toggles between Bezier, B-spline, Ellipse, and Rectangle)	Applies a Bezier shape. Bezier shapes are defined using control points and tangents.
	B-Spline	**V**	Applies a B-spline shape. Unlike Bezier shapes, B-splines are created by only using control points. The position of the points in relation to each other determines what kind of splines the shape consists of.

Icon	Tool	Hotkey	Function
○	Ellipse	V	Applies an ellipse shaped Bezier shape on the current plate.
□	Rectangle	V	Applies a rectangle shaped Bezier shape on the current plate.

Tip *You can choose to have your RotoPaint always open with a particular tool selected. If you want to open it with the Bezier tool selected, for example, do the following:*

1. Create a file called menu.py in your plug-in path directory if one doesn't already exist. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. To select Bezier as your default tool, save the following in your menu.py:

```
nuke.knobDefault('RotoPaint.toolbox','bezier')
```

3. Restart Nuke.
For more information on default tools, see "Setting Default RotoPaint Tools and Settings" on page 348.

Separate Select tools in the RotoPaint toolbar let you make changes to a stroke/shape once it's been drawn (see "Selecting Existing Strokes/Shapes for Editing" on page 351).

Using the Bezier Tool

The Bezier tool lets you draw Bezier shapes.

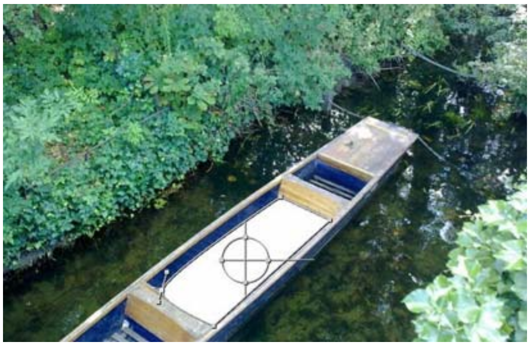


Figure 10.12: Drawing a Bezier shape.

To use the Bezier tool:

1. Click the **Bezier** in the RotoPaint toolbar.
2. Select color, blending mode, opacity, and other settings for the shape in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
3. Optionally, set the lifetime of the shape in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
4. Draw a shape in the Viewer by clicking to create the outlines that make up your shape. While drawing, you can click+drag to create a point and adjust its tangent handles. With the tangent handles, you can adjust the spline of your shape.
    - You can move the individual handles to adjust their length, keeping the angle consistent.
    - Press **Shift** while moving the tangent handles to move both handles at the same time, keeping the angle consistent.
    - Press **Ctrl/Cmd** to temporarily break the angle.
5. You can also use other shortcuts to adjust your shape while drawing:
    - **Shift**+click to create a sharp exit point on the previous point.

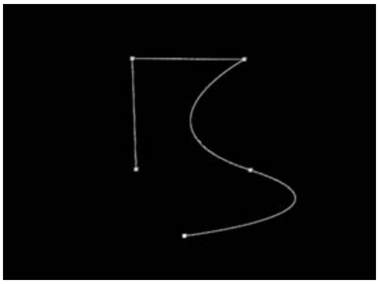


Figure 10.13: Curved exit point.

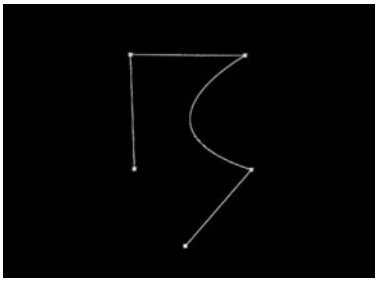


Figure 10.14: Sharp exit point.

- **Ctrl/Cmd**+click to sketch your shape freely.

6. To close your shape, press **Return** or click the first point of your shape. Changing to a different tool also closes a shape. By default, closing a shape activates the Select tool.

   If you close your shape by clicking the first point of it, you can also drag the point to create tangent handles for adjusting it.
7. With the Select tool active, you can **Shift**+click your shape points to bring up the transform handle box, which you can use to further transform your shape or particular points in your shape.

Tip *You can also apply animation from a Tracker node to a point in your Bezier shape. From the Tracker node,* ***Ctrl/Cmd****+drag the transformation information into the Viewer, on the point you want to animate.*

Using the B-Spline tool

The **B-spline** tool lets you draw B-spline shapes.

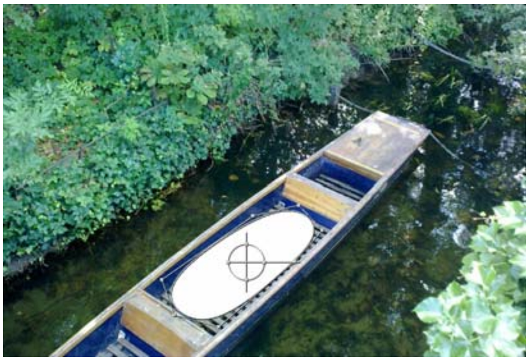


Figure 10.15: Drawing a B-spline shape.

To use the B-Spline tool:

1. Right-click the **Bezier** tool in the RotoPaint toolbar and select **B-Spline** tool.
2. Select color, blending mode, opacity, and other settings for the shape in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
3. Optionally, set the lifetime of the shape in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
4. Click in the Viewer to create the splines that make up your shape.
    - **Ctrl/Cmd**+drag to sketch the spline freely.
    - Click+drag the cursor left or right to adjust the tension (that is, the distance between the spline line and a point) on the previous point.
5. To adjust the tension of any point in the b-spline shape you're drawing, select a point, press **Ctrl/Cmd+Shift,** and drag left or right: right to increase the tension, left to decrease it.
6. To close your shape, press **Return**. Changing to a different tool also closes a shape. By default, closing a shape activates the Select All tool.
7. With the Select All tool active, you can **Shift**+click your shape points to bring up the transform handle box, which you can use to further transform your shape or particular points in your shape.

8. You can also cusp and smooth a point in your B-spline shape to create a sharp corner on your shape or smooth it again. To do this, right-click on a point in your B-spline shape and select **cusp** or **smooth**.

Tip *You can convert your B-spline shape into a Bezier shape after creating it. Select the B-spline shape using Select All tool, right-click on it and select **convert bspline to bezier**. You can now edit your shape in the same way as other Bezier shapes.*

Using the Ellipse and Rectangle Tools

The **Ellipse** and **Rectangle tools** let you draw an ellipse or rectangle shaped Bezier shape. After creation, ellipses and rectangles can be edited like normal Bezier shapes.

Figure 10.16: Drawing ellipses and rectangles.

To use the Ellipse and Rectangle tools:

1. Right-click the **Bezier** tool in the RotoPaint toolbar and select **Ellipse** tool or **Rectangle** tool.
2. Select color, blending mode, opacity, and other settings for the shape in the RotoPaint tool settings at the top of the Viewer. (For information on the available options, see "Editing Existing Stroke/Shape Attributes" on page 353.)
3. Optionally, set the lifetime of the shape in the RotoPaint tool settings. (For information on the available options, see "Editing Existing Stroke/Shape Timing" on page 362.)
4. Click+drag across the Viewer to draw an ellipse or a rectangle shape.

   If you want to create a perfect circle with the **Ellipse** tool or a square with the **Rectangle** tool, hold down **Shift** while drawing your shape.

   If you want to draw a shape from the center out, press **Ctrl/Cmd+Shift** as you're dragging.

Setting Default RotoPaint Tools and Settings

You can choose to have your RotoPaint always open with a particular tool selected, or open a particular tool with the settings you want.

To set a tool as your default tool:

1. Create a file called menu.py in your plug-in path directory if one doesn't already exist. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Select your default tool from the following list and add the Python line in your menu.py:
   - Select All
     ```
     nuke.knobDefault('RotoPaint.toolbox','selectAll')
     ```
   - Select Splines
     ```
     nuke.knobDefault('RotoPaint.toolbox','selectCurves')
     ```
   - Select Points
     ```
     nuke.knobDefault('RotoPaint.toolbox','selectPoints')
     ```
   - Select Feather Points
     ```
     nuke.knobDefault('RotoPaint.toolbox','selectFeatherPoints')
     ```
   - Add Points
     ```
     nuke.knobDefault('RotoPaint.toolbox','addPoints')
     ```
   - Remove Points
     ```
     nuke.knobDefault('RotoPaint.toolbox','removePoints')
     ```
   - Cusp Points
     ```
     nuke.knobDefault('RotoPaint.toolbox','cuspPoints')
     ```
   - Smooth Points
     ```
     nuke.knobDefault('RotoPaint.toolbox','curvePoints')
     ```
   - Remove Feather
     ```
     nuke.knobDefault('RotoPaint.toolbox','removeFeather')
     ```
   - Open/Close Curve
     ```
     nuke.knobDefault('RotoPaint.toolbox','closeCurve')
     ```
   - Bezier
     ```
     nuke.knobDefault('RotoPaint.toolbox','createBezier')
     ```
   - B-Spline
     ```
     nuke.knobDefault('RotoPaint.toolbox','createBSpline')
     ```
   - Ellipse
     ```
     nuke.knobDefault('RotoPaint.toolbox','createEllipse')
     ```

  - Rectangle

    ```
    nuke.knobDefault('RotoPaint.toolbox','createRectangle')
    ```
  - Brush

    ```
    nuke.knobDefault('RotoPaint.toolbox','brush')
    ```
  - Eraser

    ```
    nuke.knobDefault('RotoPaint.toolbox','eraser')
    ```
  - Clone

    ```
    nuke.knobDefault('RotoPaint.toolbox','clone')
    ```
  - Reveal

    ```
    nuke.knobDefault('RotoPaint.toolbox','reveal')
    ```
  - Dodge

    ```
    nuke.knobDefault('RotoPaint.toolbox','dodge')
    ```
  - Burn

    ```
    nuke.knobDefault('RotoPaint.toolbox','burn')
    ```
  - Blur

    ```
    nuke.knobDefault('RotoPaint.toolbox','blur')
    ```
  - Sharpen

    ```
    nuke.knobDefault('RotoPaint.toolbox','sharpen')
    ```
  - Smear

    ```
    nuke.knobDefault('RotoPaint.toolbox','smear')
    ```

3. Restart Nuke.

To set your default tool settings:

1. Create a file called init.py in your plug-in path directory if one doesn't already exist. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Define your RotoPaint tool and the default setting you want to set. To get the specific tool and setting names, you can copy your RotoPaint node in Nuke and paste it into a text editor. The lines with curved brackets after "toolbox" indicate your current tool settings.

- For example:
  - to set the **brush size** for paint to **30**, and for **clone** to **280** , add this Python line in your init.py:

    ```
    nuke.knobDefault("RotoPaint.toolbox", '''clone {
      { brush bs 30 }
    ```

```
  { clone bs 280 }
}''')
```

OR

- to set the **brush hardness** for paint to **1.0** by default, add this Python line:

  ```
  nuke.knobDefault("RotoPaint.toolbox", '''brush {
    { brush h 1 }
  }''')
  ```

- to set the **source transform** round on clone to **on** by default, add this Python line:

  ```
  nuke.knobDefault("RotoPaint.toolbox", '''clone {
    { clone str 1 }
  }''')
  ```

3. Restart Nuke.

Selecting the Output Format and Channels

In the RotoPaint control panel, you can select one output channel or many to indicate the channels where the results of your changes should be stored. If you have no input connected, select an output format using **format**.

1. From the **output** field, select the channel set containing the channels you want to use. By default, **rgba** is selected, and the red, green, blue, and alpha channels are checked on the right.
2. Uncheck any channels that you don't want to process. The node will process all those you leave checked. For more information on selecting channels, see "Calling Channels" on page 174.
3. If you want to, you can use the **output mask** drop-down to select a channel where RotoPaint will output a mask for what it rendered. By default, the channel is **none**, but if you choose a channel in the list, the **output mask** box will be checked.

   The mask can be useful, for example, if you need to apply grain to the areas you've painted, but you don't want to double up the grain in other areas.
4. If necessary, choose your **premultiply** value.

   Premultiply multiplies the chosen input channels with a mask representing the paint strokes and shapes. For example, where there are no paint strokes or shapes (the paint matte is black or empty) the input channels will be set to black, and where the paint strokes or shapes are opaque (the paint matte is white or full) the input channels keep their full value.

Note that selecting **rgba** premultiplies the alpha against itself (a*a). If you don't want this to happen, set **premultiply** to **rgb** instead.

5. From the **clip to** menu, select how you want to restrict the output image:
   - **no clip** - Do not restrict the output image.
   - **bbox** - Restrict the output image to the incoming bounding box
   - **format** - Restrict the output image to the incoming format area (the default).
   - **union bbox+format** - Restrict the output image to a combination of the bounding box and the incoming format area.
   - **intersect bbox+format** - Restrict the output image to an intersection of the bounding box and incoming format area.

   You can also check the **Replace** box if you want to clear the channels to black before drawing into them. You might find **Replace** useful, for instance, if you're creating a mask in the alpha channel, but the incoming image already has an alpha channel which you want to throw away.
6. If necessary, choose your **format** value.

   Format is used if RotoPaint has no input connected. It is the format which the node should output in the absence of any available input format. If an input is connected, this control has no effect.

Selecting Existing Strokes/Shapes for Editing

If you've already drawn a stroke/shape but wish to make changes to it, you can select it or certain parts of it with the RotoPaint selection tools. You can also toggle between all of them in the RotoPaint toolbar using the shortcut **Q**.

You can also use the controls in the RotoPaint tool settings to display and hide information such as point numbers (see "Viewing Point Numbers" on page 352) and paint stroke splines (see "To select an entire stroke/shape:" on page 351).

To select an entire stroke/shape:

1. Click the **Select All** tool, or press the hotkey **Q**.
2. Select the stroke/shape you wish to edit either by clicking on it in the Viewer or by clicking on its name in the stroke/shape list. To select several strokes/shapes, **Ctrl**/**Cmd**+click or **Shift**+click (to select a range) their names in the stroke/shape list.

   When selecting strokes/shapes in the Viewer, you can invert your selection by right-clicking and selecting **invert selection**. All strokes/shapes you didn't have selected before are now selected.

Tip *To select the position of a paint stroke and view your selection in the Viewer, you have to check the **allow paint selection** box in the tool settings bar.*

To select a spline:

1. Right-click the **Select All** tool and select the **Select Splines** tool.
2. Select the spline you wish to edit either by clicking on it in the Viewer or by clicking on its name in the stroke/shape list. Selecting a spline only selects the spline, not points within it.

Tip *Using the **Select Splines** tool, you can also duplicate the stroke/shape you've selected. Just right-click on one of the points, and select **duplicate curve**. A new stroke/shape is created with the same spline and attributes as the one you selected.*

To select only points on a stroke/shape:

1. Right-click the **Select All** tool and select the **Select Points** tool.
2. Select the stroke/shape you wish to edit by clicking on its name in the stroke/shape list and then select a point in the Viewer. To select several points, **Ctrl**/**Cmd**+click on them in the Viewer or use marquee selection to create a transform handle box.

   Using **Select Points** tool restricts selection to one stroke/shape only.

To select points using a transform handle:

1. Select the **Select All** tool from the RotoPaint toolbar.
2. Select **show transform handle** in the RotoPaint tool settings.
3. Select points in a shape/stroke with **Shift**+click or by clicking and dragging across the points you want to select. A transform handle box appears.
4. You can also use shortcut **T** to toggle viewing the transform handle.

Viewing Point Numbers

You can view the numbers for points in your stroke/shape in the Viewer. With a Select tool activated, check the **label points** box in the RotoPaint tool settings. Feather points are marked with a bracketed number corresponding their stroke/shape point, so for example, if a stroke/shape point is marked with the number 2, its feather point is marked with [2].

You can also copy point links and use them in other nodes, for example. To copy a point link:

1. Right-click on a point
2. Select **copy** > **point link**.
3. Go to the field you want to use the link in, for example another node, right-click and select **Copy** > **Paste**.

Editing Existing Stroke/Shape Attributes

After selecting a stroke/shape using the stroke/shape list or the one of the Select tools, you can edit and animate their attributes in the control panel. For more information on selecting strokes/shapes, see "Selecting Existing Strokes/Shapes for Editing" on page 351 and "Working with the Stroke/ Shape List" on page 331.

If you want to edit the attributes of a stroke/shape prior to drawing one, you should do that in the RotoPaint tool settings in the Viewer. (See "Working with the Toolbars" on page 331.)

Editing Attributes Common to Strokes and Shapes

Many controls in the RotoPaint control panel apply to both strokes and shapes. These controls are grouped under the **RotoPaint** tab.

Editing color

When drawing a stroke/shape (see "Using the Brush tool" on page 335, "Using the Bezier Tool" on page 344, and "Using the B-Spline tool" on page 346), you can set the RGBA color values of the stroke/shape using the **color** controls on the **RotoPaint** tab of the RotoPaint control panel (seeChapter 3, "Using the Color Picker and Color Controls", on page 65). You can also adjust color directly in the stroke/shape list using the control in the **color** column.

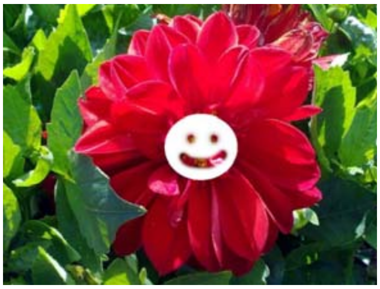


Figure 10.17: **Color** set to white (the default).

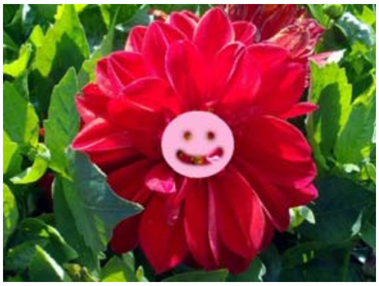


Figure 10.18: **Color** set to pink.

Editing opacity

You can set the opacity of the stroke/shape using the **opacity** slider. If you set the opacity of a shape to zero, the outline for it won't be drawn unless the shape is selected. You can also temporarily make the stroke/shape invisible (that is, completely transparent) by toggling the **visible** box in the stroke/shape list.

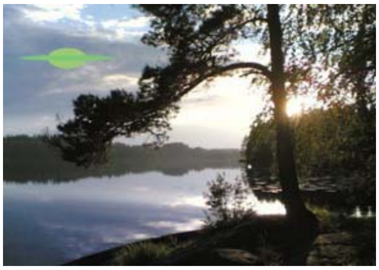


Figure 10.19: A low **opacity** value.

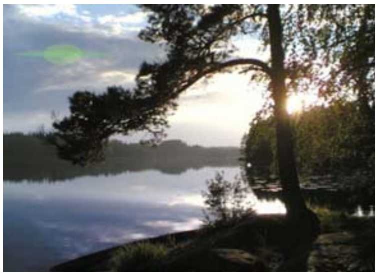


Figure 10.20: A high **opacity** value.

When drawing brush strokes, you can tie their transparency to pen pressure. Just check the **opacity** box next to **pressure alters** in the control panel.

Selecting a source for your stroke/shape

You can choose a source for your stroke/shape and define whether it's a color, a background or a foreground image. With your stroke/shape selected, choose a source for your stroke/shape from the **source** drop-down on the **RotoPaint**, **Stroke** or **Shape** tabs.

Editing blending mode

By choosing different blending modes from the **blending mode** dropdown in the control panel, you can choose how the colors in your strokes/shapes blend with the underlying image. You can also apply blending modes to your strokes, shapes, or groups directly in the stroke/shape list using the **blending mode** column.

Each of the blending modes blends the primary color, that is the color of the current stroke/shape/group you're editing with the secondary color, which is the combined color of your previously rendered strokes/shapes/groups.

The different modes are as follows:

- **Color burn** - Darkens the primary color to reflect the secondary color by increasing the contrast. No part of the image will become lighter.

- **Color dodge** - Brightens the primary color to reflect the secondary color by decreasing the contrast. No part of the image will be darkened.
- **Difference** - Subtracts either the secondary color from the primary color or vice versa, depending on which is brighter. Blending with white inverts the primary color, while blending with black produces no change. Similar colors will return black pixels. Difference is a useful mode when working with mattes.
- **Exclusion** - Creates a result similar to the Difference mode but lower in contrast. Like with Difference, blending with white inverts the primary color. Blending with black produces no change.
- **From** - Subtracts the primary color from the secondary color.
- **Hard Light** - Lightens highlights and darkens shadows. If the secondary color is lighter than 50% gray, the result lightens as if it were screened. If the secondary color is darker than 50% gray, the result is darkened as if it were multiplied.
- **Max** - Selects the lighter of the two colors as the resulting color. Only areas darker than the secondary color are replaced, while areas lighter than the secondary color do not change.
- **Min** - Selects the darker of the two colors as the resulting color. Any parts that are lighter than the secondary color are substituted. Any parts of the image that are darker than the secondary color don't change.
- **Minus** - Subtracts the secondary color from the primary color.
- **Multiply** - Multiplies the primary color by the secondary color. The result is always darker. Blending with black gives black and with white returns the color unchanged.
- **Over** - This mode is the default. The colors of the two images will not interact in any way, and Nuke will display the full value of the colors in the primary image.
- **Overlay** - Depending on the primary color, multiplies or screens the colors. The secondary color brightens the primary color while preserving highlights and shadows.
- **Plus** - The sum of the two colors. Increases brightness to lighten the primary color and reflect the secondary color. Plus is similar to the Screen blending mode, but produces a more extreme result.
- **Screen** - This is a soft Plus making everything brighter but ramping off the whites. Light colors have more of an effect than dark colors. The result is always a lighter color. Blending with black leaves the pixel unchanged, blending with white always returns white. The result is similar to projecting multiple slides on top of each other.
- **Soft Light** - Depending on the primary color, darkens or lightens the colors. Less extreme than the Hard Light mode.

Tip *Note that changing the stack order of your primary and secondary colors might have an impact on your result. For example, if you have two Bezier shapes overlapping each other with a blending mode active, the result will depend on which shape is on top of the other. You can change the stack order of strokes/shapes in the stroke/shape list.*

Transforming Strokes/Shapes/Groups

To apply spatial transformations to your strokes, shapes, or groups, you can use the controls under the **Transform** tab. Select a stroke/shape/group from the stroke/shape list and adjust:

- **translate** - to move the stroke/shape on x and y axis.
- **rotate** - to spin a stroke/shape around the pivot point. Use **center** to position the pivot point.
- **scale** - to resize a spline. Use **center** to position the pivot point.
- **skew** - to rotate the spline of your stroke/shape around the pivot point. Use **center** to position the pivot point.
- **extra matrix** -

Alternatively, you can also use the transform handle (shortcut **T**) in the Viewer to transform elements. To transform an entire stroke/shape, you'll need to use the transform handle jack, and to transform points in a stroke/shape, you should use the transform handle box.

The transform handle appears as a transform jack only when the **Transform** tab is active, when any of the other tabs in the RotoPaint control panel are active, the transform handle appears as a box.

To transform a stroke/shape using a transform handle jack:

1. Click the **Select All** tool in the RotoPaint toolbar, with the **Transform** tab active.
2. Select **show transform handle** in the RotoPaint tool settings.
3. Select a stroke/shape with by clicking it in the Viewer or by selecting it in the stroke/shape list. A transform handle jack appears.
4. Use the jack for instance to rotate, scale, skew your stroke/shape.

To transform points using a transform handle box:

1. Click **Select All** tool or **Select Points** tool in the RotoPaint toolbar, with the RotoPaint tab active.
2. Select **show transform handle** in the RotoPaint tool settings.

3. Select certain points in a stroke/shape with **Shift**+click or by clicking and dragging across the points you want to select. A transform handle box appears.
4. Use the box for instance to rotate, scale, skew your stroke/shape, or points.
5. To corner pin using the transform box, press **Ctrl/Cmd+Shift** and drag the transform box points to move them.

Tip *Transforming points changes the actual point position, transforming an entire stroke/shape/group changes transformation applied to the point positions.*

To transform onion skin source

When cloning or revealing, you can use the onion skin control on the RotoPaint tool settings to view and transform your source input on top of your foreground. You can also use onion skinning if you're drawing a stroke/shape with a separate input as the source. To adjust onion skin:

1. With your stroke/shape tool selected, check the **onion skin** box in the RotoPaint tool settings.
2. Adjust the opacity of the onion skin and transform your souce by using the onion skin transform overlay in the Viewer.

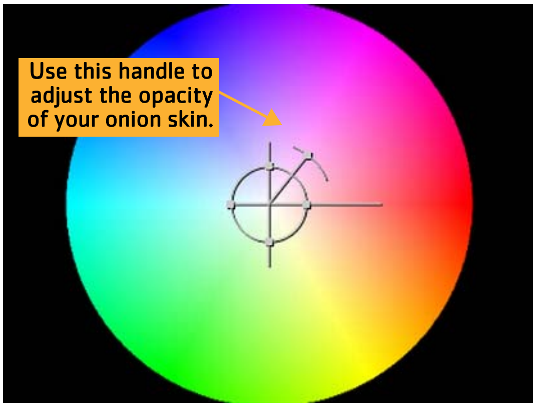


Figure 10.21: Onion skin transform handle

Adjusting Mask Controls

By default, **mask** is set to **none**, but if you want to use a mask, do the following:

1. Check the **mask** checkbox on the **RotoPaint** tab.
2. Select a channel from the dropdown menu.

3. If you are using the mask input and want the mask copied into the predefined mask.a channel, check **inject**. This way, you can use the mask input again downstream.
4. If necessary, check **invert** to reverse the mask and/or **fringe** to blur the edges of the mask.
5. If you find that the overall effect of the RotoPaint node is too harsh, you can also blend some of the original input image back in by adjusting the **mix** slider.

Editing Shape Specific Attributes

The RotoPaint control panel includes a set of controls that you mainly need when you're editing the attributes of a shape. You can find these, in the **Shape** tab in the control panel.

Adding and removing feather

To soften the edges of a shape, do the following:

1. With your shape selected, check the **on** box next to the feather slider on the **Shape** tab to apply feather to your shape. If you don't want any feather on your shape, uncheck the **on** box.
2. Use the **feather** slider to add outward or inward feather around the whole shape. With positive feather values, your feather effect is outward and, respectively, if your feather values are negative, the feather effect is inward.
3. Use the **feather falloff** slider to adjust the falloff profile. Falloff is measured in pixels. You can also change the type of the falloff using the **falloff type** dropdown list. Choose between **linear**, **smooth0**, **smooth1** and **smooth**. Each of these produces a different rate of falloff that may be helpful for example in matching the soft edge to motion blurred image content.
4. To add feather to a single point, right-click on the point in the Viewer and select **increase feather**. The shortcut for this is **E**, and if you press it several times, every press adds more feather.
5. Use the feather handle on the point to add feather into the point. By default, the point angle is locked, and moving the point unlocks the angle.

   You can also select several feather points and move them together.
6. To remove feather from a point, right-click on the point and select **reset feather** or use the shortcut **Shift+E.**
7. If you deselect the **feather link** box (selected by default) in the RotoPaint tool settings, the feather effect doesn't move if you move the shape points.

Adding motion blur to a shape

1. With your shape selected, check the **on** box next to the **motionblur** field to apply motion blur to your shape. If you don't want any motion blur on your shape, uncheck the **on** box.
2. In the **motionblur** field, enter the sampling rate. This affects the number of times the input is sampled over the shutter time. The higher the rate, the smoother the result. In many cases, a value of 1.0 is enough.

   You can also add motion blur to shapes in the stroke/shape list using the **Motionblur** column.
3. In the **shutter** field, enter the number of frames the shutter stays open when motion blurring. For example, a value of 0.5 would correspond to half a frame. Increasing the value produces more blur, and decreasing the value less.
4. Adjust the shutter offset using the **shutter offset** dropdown menu. The different options control when the shutter opens and closes in relation to the current frame value. Select:
   - **centered** - to center the shutter around the current frame. For example, if you set the shutter value to 1 and your current frame is 30, the shutter will stay open from frame 29,5 to 30,5.
   - **start** - to open the shutter at the current frame. For example, if you set the shutter value to 1 and your current frame is 30, the shutter will stay open from frame 30 to 31.
   - **end** - to close the shutter at the current frame. For example, if you set the shutter value to 1 and your current frame is 30, the shutter will stay open from frame 29 to 30.
   - **custom** - to open the shutter at the time you specify. In the field next to the dropdown menu, enter a value (in frames) you want to add to the current frame. To open the shutter before the current frame, enter a negative value. For example, a value of - 0.5 would open the shutter half a frame before the current frame.

Editing Stroke Specific Attributes

The RotoPaint control panel includes a set of controls that you mainly need when you're editing the attributes of a paint stroke. You can find most of these under the **Stroke** tab, in the control panel.

Selecting a source image

On the **RotoPaint** tab, you can set the **source** control to a specific color or the input you want to pull pixels from for Clone and Reveal brushes. Choose:

- **color** - to use a specific color in your stroke/shape.

- **foreground** - to pull pixels from the RotoPaint's **bg** input, including any strokes/shapes drawn on it. This input is mainly used with cloning.
- **background** - to pull pixels from the **bg** input, not including any strokes/shapes drawn on it.
- **background 1**, **background 2** or **background 3** - to pull pixels from the **bg1**, **bg2,** or **bg3** input.

Editing brush type

On the **Stroke** tab, you can choose the type of brush you want to use for the stroke. Select:

- **paint** - to use a normal paint brush.
- **smear** - to use a smear brush on the plate.
- **blur** - to blur your plate with the brush stroke.
- **sharpen** - to sharpen your plate with the brush stroke.

Editing brush size

On the **Stroke** tab, you can set the size of the stroke using the **brush size** slider. You can also tie a stroke's size to pen pressure by checking the **size** box next to **pressure alters** in the RotoPaint control panel.

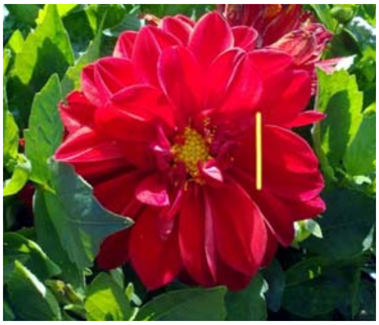


Figure 10.22: A low **brush size** value.

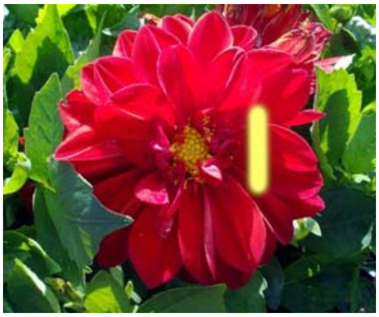


Figure 10.23: A high **brush size** value.

Editing brush spacing

The **brush spacing** slider adjusts the distance in pixels between paint brush dabs. A higher setting will increase the space between dabs, creating a dotted line effect when painting. A lower setting will decrease the distance and create a solid brush stroke.

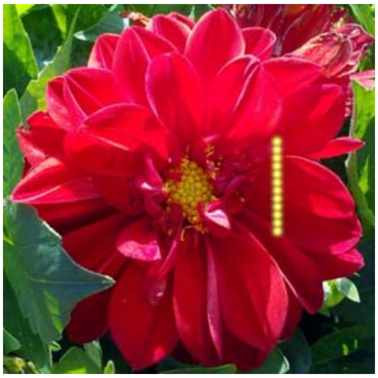


Figure 10.24: A low **brush spacing** value.

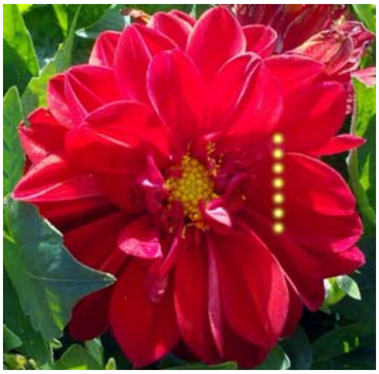


Figure 10.25: A high **brush spacing** value.

Editing brush hardness

On the **Stroke** tab, you can set the hardness of the stroke using the **brush hardness** slider.

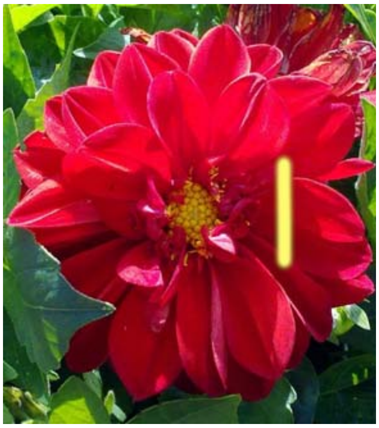


Figure 10.26: A low **brush hardness** value.

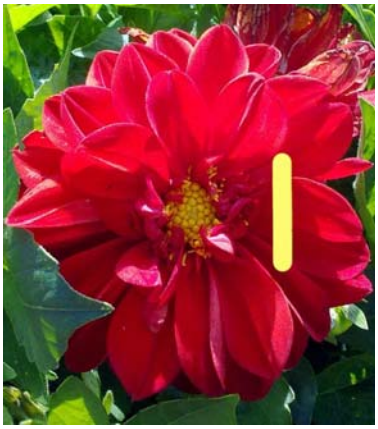


Figure 10.27: A high **brush hardness** value.

You can also tie a stroke's hardness to pen pressure by checking the **hardness** box next to **pressure alters**.

Adjusting write on

When you are animating a stroke or a part of it over a range of frames, you can use the **write on** sliders under the **Stroke** tab in the control panel to adjust the order in which the dabs on the stroke appear over these frames.

For more information on animating parameters, see "Animating Parameters" on page 71.

- **write on start** - slide to choose where along the stroke length the paint begins. 0 is the start of the stroke, 1 is the end.
- **write on end** - slide to choose where along the stroke length the paint ends.

Editing Clone or Reveal Attributes

When you are using the Clone or Reveal tool, you can adjust the controls under the **Clone** tab to transform the input that's being cloned or revealed. Adjust:

- **translate** - to move the source image on x and y axis.
- **rotate** - to spin the source image around a pivot point.
- **scale** - to resize the source image by adding or removing pixels. Use **center** to position the pivot point.
- **skew** - to rotate the pixel columns of the source image around the pivot point. Use **center** to position the pivot point.
- **filter** - to choose the appropriate filtering algorithm. For more information, see "Choosing a Filtering Algorithm" on page 227.
- **black outside** - When rotating or translating the clone source, a part of the image area may get cropped. To fill the cropped portion with black, check **black outside**. To fill the cropped portion by expanding the edges of the image, uncheck **black outside**.
- **time offset** - to clone or reveal pixels from a different frame. Time offset is either relative to the current frame (-1 is the frame previous to the current one) or absolute (1 is the first frame in the clip).
- **view** - to choose which view you want to clone from in a stereoscopic project.

Editing Existing Stroke/Shape Timing

When editing an existing stroke/shape, you can edit the range of frames during which a stroke/shape is visible. The lifetime of a stroke/shape/group is also visible in the **Life** column in the stroke/shape list. By default, a shape is visible on all frames, whereas a stroke is only visible on one frame, the frame it was painted on.

To make a stroke/shape visible for all frames (the default):

Under **lifetime type**, select **all frames** or press the **all frames** button.

To make a stroke/shape visible from the current to the last frame:

Under **lifetime type**, select **frame to end** or press the **frame to end** button.

To make a stroke/shape visible only on the current frame:

Under **lifetime type**, select **single frame** or press the **single frame** button.

To make a stroke/shape visible from the first frame to the current frame:

Under **lifetime type**, select **start to frame** or press the **start to frame** button.

To make a stroke/shape visible during a specified range of frames:

1. Under **lifetime type**, select **frame range** or press the **frame range** button. A dialog box prompts for the frame range.
2. Enter the start and end frames for the range during which you want the stroke to appear, separated by a hyphen (-). Click **OK**.

Editing Existing Stroke/Shape Stack Order

When editing strokes/shapes after you've drawn them, you can edit their foreground to background drawing order.

In the stroke/shape list, you can drag and drop strokes/shapes to change their drawing order, and to group them under folders. For more information on using the stroke/shape list, see "Working with the Stroke/Shape List" on page 331.

Editing Existing Stroke/Shape splines

To edit a stroke/shape position, you first need to select the stroke/shape in the Viewer or the stroke/shape list. You can then modify the points that make up the stroke/shape position.

To add a point to a position:

1. Select the **Add Points** tool from the RotoPaint toolbar.
2. In the Viewer, click on the selected stroke/shape to add a new point.

   You can also add a point by pressing **Ctrl/Cmd+Alt** and click on a selected stroke/shape.

To move a point:

1. Select the stroke/shape in the Viewer or the stroke/shape list.
2. With the **Select All** tool or **Select Points** tool active, in the Viewer, drag the points you want to move to a new location.
3. You can also nudge a point using the arrows on your numeric keypad. Nudging a point moves it by one pixel to the direction you choose.

Tip *If you find it difficult to select a single point, you might want to make the* ***handle size*** *or the* ***handle pick size*** *larger in the Nuke Preferences dialog* ***(Preferences > Viewers > Handles).***

To move several points together:

1. With the **Select All** tool or **Select Points** tool active, select the stroke/shape in the Viewer or the stroke/shape list.
2. Select **Show transform handle** in the tool settings, if it's not already selected.
3. In the Viewer, drag a marquee around the points (or an entire stroke/shape) that you want to move. A transform handle box appears.
4. Adjust the transform handle as necessary.

To delete a point:

1. Right-click on the **Add Points** tool and select **Remove Points** tool.
2. Select the stroke/shape in the Viewer or the stroke/shape list.
3. In the Viewer, click the point that you want to delete.

   OR

1. Select the stroke/shape in the Viewer or the stroke/shape list.
2. In the Viewer, right-click on the point that you want to delete and select **delete**.

   OR

1. Select the stroke/shape in the Viewer or the stroke/shape list.
2. Click the point you want to delete and press the **Delete/Backspace** key.

To delete an entire stroke/shape:

1. In the stroke/shape list, or in the Viewer with the **Select All** tool, click on the stroke/shape you want to delete.
2. Below the stroke/shape list, click on the minus button (-),

   OR

1. Activate the **Select All** tool. . In the **Viewer**, click on the stroke/shape.
2. Right-click on the shape and select **delete** or press the **Delete**/**Backspace** key.

To cusp or smooth points:

You can cusp points on a shape to create sharp corners, and smooth points to replace sharp corners with curved lines.

1. Select the shape you want to edit in the Viewer or the stroke/shape list.
2. Select the **Smooth Points** tool or **Cusp Points** tool in the RotoPaint toolbar, depending on whether you want to cusp or smooth your points.
3. In the Viewer, click on the point that you want to cusp or smooth.
4. With the tangent handles, adjust the shape of your angle.
5. With you point selected, you can also use the shortcut keys **Z** and **Shift+Z** to smooth and cusp it respectively.

   OR

   Right-click on the point you want to smooth or cusp, and select **smooth** or **cusp**.

To add expressions to points:

1. With the **Select All** tool or **Select Points** tool active, select a point in your stroke/shape.
2. Right-click and select **add expression**.
3. Enter your expression values to the fields in the Expression dialog. Alternatively, you can **Ctrl/Cmd**+drag expression values from another node on the point.

   You can add expressions to edit your point's location or the shape of the point's feather effect. For more information on Expressions see "Expressions" on page 533.

Animating Strokes/Shapes

All strokes/shapes that appear on more than one frame can be animated. By default, the **autokey** option is on, which means your changes to a stroke/shape will automatically create keyframes and animate your stroke/shape. You can also access all the curves and shapes in the Curve Editor.

To animate a stroke/shape using autokey:

1. Draw a stroke/shape that appears on more than one frame. By default, the **autokey** option in the RotoPaint tool settings is selected and a keyframe is automatically created in the first frame your stroke/shape appears.
2. Move to a new frame.
3. With one of the Select tools, select the points or the stroke/shape you want to animate.
4. Adjust the points in your stroke/shape position or change the stroke/shape's attributes as necessary. A new keyframe is automatically set. The frame marker on the timeline turns blue to indicate the selected stroke/shape is animated.
5. Repeat steps 2, 3 and 4 for all the frames you want to set as key frames.

Tip *Note that if you are translating an entire stroke/shape, RotoPaint will also draw a track of the stroke/shape's animated position, which you can view by activating the* ***Transform*** *tab. You can, for example, use the track in another node (such as Tracker or CameraTracker) by* ***Cmd/Ctrl****+dragging the values from the translate animation button to an appropriate field in another node.*

To view spline keyframes for a shape/stroke

You can use the **spline key** controls on the **RotoPaint** tab to view whether there are keyframes set for the spline of your stroke/shape. Do the following:

1. Select a stroke/shape on the stroke/shape list.
2. If there are spline keys set on your stroke/shape, the key boxes are highlighted blue and display where you are currently on the timeline with regard to the keyframes that are set already. You can move backwards and forwards between the keyframes using the arrow keys.
3. If you want to add or remove spline keys for the stroke/shape in a selected frame, use the **Add** and **Delete** buttons.

To animate strokes/shapes manually:

If you choose to switch the **autokey** function off, you can still create key frames manually. You can set key frames to the entire stroke/shape, or the stroke/shape's spline, transformation or attributes.

1. Move to the frame where you want to create a keyframe and select your stroke/shape.

2. Do one of the following:
    - If you want to create a key that is set to animate the entire stroke/ shape, right-click on the stroke/shape and select **set key > all**.
    - If you want to create a key that is set to animate a position, right-click on the stroke/shape and select **set key > shape.**
    - If you want to create a key that is set to animate transformation, right-click on the stroke/shape and select **set key > transform**.
    - If you want to create a key that is set to animate attributes, right-click on the stroke/shape and select **set key > attributes**.

    If you have autokey turned off, you can only adjust a point in a shape/ stroke at a keyframe. In other words, in order to make changes to a point, you either have to move to an existing keyframe on the timeline, or you need to create a new keyframe first.

To view keyframes on the timeline:

You can view different types of keyframes you've set, either automatically or manually, on the timeline. If you've set keyframes on the **RotoPaint** tab, these will be visible on the timeline when you have the **RotoPaint** tab open in the control panel. Similarly, if you've created transformation keyframes on the **Transform** tab, you can see those keyframes on the timeline when you have the **Transform** tab open.

To delete a keyframe:

1. Using the **Viewer** timeline, scrub to the frame where you want to delete a keyframe.
2. In the stroke/shape list, select the stroke/shape whose key you want to delete.
3. Do one of the following:
    - If you want to delete a key that is set to animate the entire stroke/ shape, right-click on the stroke/shape and select **delete key > all.**
    - If you want to delete a key that is set to animate a position, right-click on the stroke/shape and select **delete key > shape.**
    - If you want to delete a key that is set to animate a transformation, right-click on the stroke/shape and select **delete key > transform.**
    - If you want to delete a key that is set to animate attributes, right-click on the stroke/shape and select **delete key > attributes**.

If no other keys are set for the selected stroke/shape, the frame marker on the timeline turns from blue to the default color.

To delete all key frames for a stroke/shape:

You can also delete all key frames you've set for a stroke/shape at one go. Do the following:

1. Select the stroke/shape from which you want to delete key frames in the Viewer.
2. Right-click on the stroke/shape and select **no animation** > **all**, **spline**, **transform** or **attribute** depending on whether you want to delete all key frames for that stroke/shape or only key frames animating shape, transform or attributes. All key frames are removed from the stroke/shape you selected.

To ripple keyframes:

Rippling keyframes allows you to adjust the position of a stroke/shape point on one frame and have that same relative adjustment applied to the point across all frames or a specified range of frames. This way, you can make non-animated changes to a stroke/shape which is being animated over a set of frames.

1. Activate **Select All** tool in the RotoPaint toolbar and check the **ripple edit** box in the RotoPaint tool settings. A red border displays around the Viewer to indicate that the ripple mode is active. In the drop-down menu, select:
   - **all** - to ripple all frames in your sequence.
   - **from start** - to ripple frames from the first frame to the current frame.
   - **to end** - to ripple frames from current frame to the last frame.
   - **range** - to ripple a particular range of frames.
2. Select the stroke/shape you want to edit from the stroke/shape list.
3. Make the necessary changes to the stroke/shape in the Viewer.

Copying, Pasting, and Cutting Stroke Positions

After creating a stroke/shape, you can copy, paste, and cut its position to use the same shape in other strokes/shapes.

Copying Point Positions

You can copy point position values in a stroke/shape you have selected. This enables you to use the same values for another point in a stroke/shape. Unlike cutting point values described below, copying position values does not delete any keys set.

To copy point values:

1. In the Viewer, scrub to the frame that contains the stroke/shape whose point position you want to copy.
2. Select the point in a stroke/shape using the **Select Points** tool.
3. Right-click on the point and select **copy > point values.**

   Nuke copies the positions of the selected point to the clipboard. Any keys set to animate these positions are not affected.

Pasting Point Positions

You can paste any position you've copied/cut from another point to a selected point in a stroke/shape. If you have **autokey** turned on, this will also set a keyframe at the current frame to animate this point. The attributes of the strokes or any keys set to animate the attributes are not affected.

To paste point positions:

1. In the **Viewer**, scrub to the frame that contains the stroke/shape to which you want to paste the positions on the clipboard.
2. Select the point in a stroke/shape using the **Select Points** tool.
3. Right-click on the point and select **paste** > **point values.**

   Nuke pastes the positions (but not any attributes) on the clipboard to the selected point and, if you have autokey turned on, sets the current frame as a keyframe.

Cutting Point Positions

You can cut the position values of a point in a stroke/shape. Cut also copies the positions at the current frame to the clipboard. The point attributes or any keys set to animate the attributes are not affected.

To cut point positions:

1. In the Viewer, scrub to the frame that contains the stroke/shape whose point position you want to cut.
2. Select the point in the stroke/shape using the **Select Points** tool.
3. Right-click on the point and select **cut point values.**

   Nuke deletes any keys set to animate the positions of the selected point, and copies the position to the clipboard.

RotoPaint and Stereoscopic Projects

When you are using the RotoPaint node to draw a new stroke/shape in a stereoscopic or multi-view project, you can toggle the **single view** checkbox in the RotoPaint tool settings to draw your stroke/shape on one view only or multiple views. For existing strokes/shapes/groups, you can use the **view** control to select the view the stroke/shape is visible. If you're working on a stereoscopic project, the view you're using for a particular stroke, shape, or group is also visible on the **View** column in the stroke/shape list. For more information on reproducting strokes/shapes on other views, see "Reproducing Paint Strokes, Beziers, and B-spline Shapes" on page 504.

Where Are the Bezier and Paint Nodes

The pre-6.0 Bezier and Paint nodes have been deprecated in favor of the new RotoPaint node. With RotoPaint you have the ability to add more strokes and shapes, group them, etc. However, Bezier and Paint are still in the application for backwards-compatibility with old scripts. Should you find the need (or just feel nostalgic), you can create the Paint and Bezier nodes in a couple of easy ways:

- Press **X** on the Node Graph, make sure **TCL** is checked in the dialog that opens, enter **Paint** or **Bezier** in the **Command** field, and click **OK**. Your node appears in the Node Graph.
- You can add a Paint or a Bezier node to your tool bar menu with a statement in your menu.py file like the following:
  - For Bezier:
    ```
    tb = nuke.toolbar("Nodes")
    tb.addCommand("Draw/Bezier",
    "nuke.createNode(\"Bezier\")", icon="Bezier.png")
    ```
  - For Paint:
    ```
    tb = nuke.toolbar("Nodes")
    tb.addCommand("Draw/Paint",
    "nuke.createNode(\"Paint\")", icon="Paint.png")
    ```

11 TEMPORAL OPERATIONS

This chapter explains the temporal or time-based operations in Nuke. You learn how to distort time (that is, slow down, speed up, or reverse clips), apply motion blur, and perform editorial operations like slips, cuts and splices.

Distorting Time

Time distortion changes the length of time required to playback a clip in your composite. These operations generally fall under one of two categories: *retiming* and *warping*.

Retiming is the process of slowing playback by adding frames, or accelerating playback by subtracting frames.

Warping is the process of slowing down, speeding up, or even reversing playback on a clip without necessarily altering the overall length.

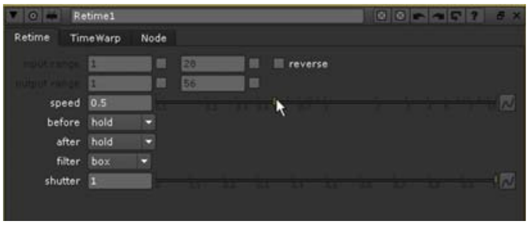
Tip *When working with temporal operations, it helps to attach a Viewer to the retiming or warping node so you can see the effect of your changes.*

Simple Retiming

Nuke's Retime node lets you change the playback time for all the frames in a clip or for range of frames within the clip. You can also use it to reverse the clip playback. It does this by dropping or duplicating frames. For higher quality retiming see "OFlow Retiming" on page 376.

To retime all frames in a clip:

1. Choose **Time** > **Retime** to insert a Retime node into your script.
2. Enter a value in the **speed** parameter. Values higher than 1 increase playback speed; values less than 1 decrease playback speed.



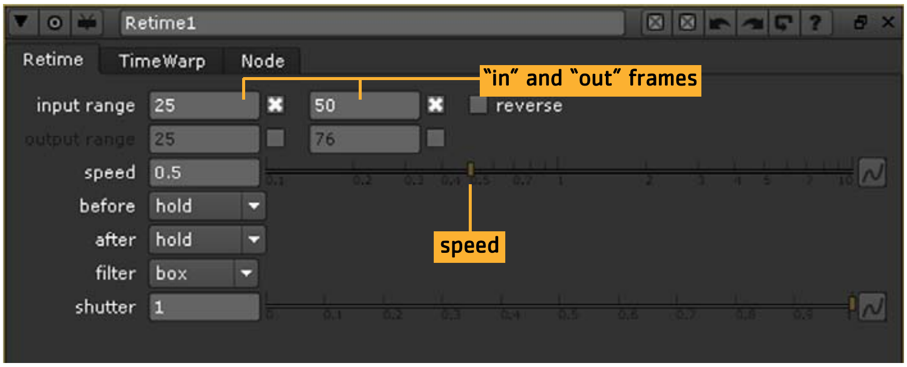
input range 1 28
output range 1 56
☐ reverse
speed 0.5
before hold
after hold
filter box
shutter 1

3. Check the **reverse** box if you want to play the clip backwards—making the last frame the first, the first frame the last, and so on.

4. Increase the **shutter** parameter to enable frame-blending (For more information, see “Interpolation” below).

To retime a range of frames in a clip:

1. Choose **Time > Retime** to insert a Retime node into your script.
2. Check the boxes for **input range** and enter the “in” and “out” frames.



input range 25 ☒ 50 ☒ ☐ reverse
output range 25 ☐ 76 ☐
speed 0.5
before hold
after hold
filter box
shutter 1

For example, if your original clip is 50 frames, but you want to only retime the last half, you would input **25** for the in point and leave the out point at **50**.

3. Check the box for **output range** and enter the “in” and “out” frames to retime to a specific clip length.

   or

   Enter a factor in the **speed** parameter and Nuke will calculate the **output range** for you. Values higher than 1 increase playback speed; values less than 1 decrease playback speed.
4. Check the **reverse** box to invert the selected frame range.
5. Increase the **shutter** parameter to enable frame-blending.

Interpolation

Time distortions that slow down a clip require the creation of additional, or *interpolated*, frames. For example, suppose you want to slow down the collision, shown in the clip below, by a factor of two. This requires the creation of one interpolated frame for every existing frame in the clip.

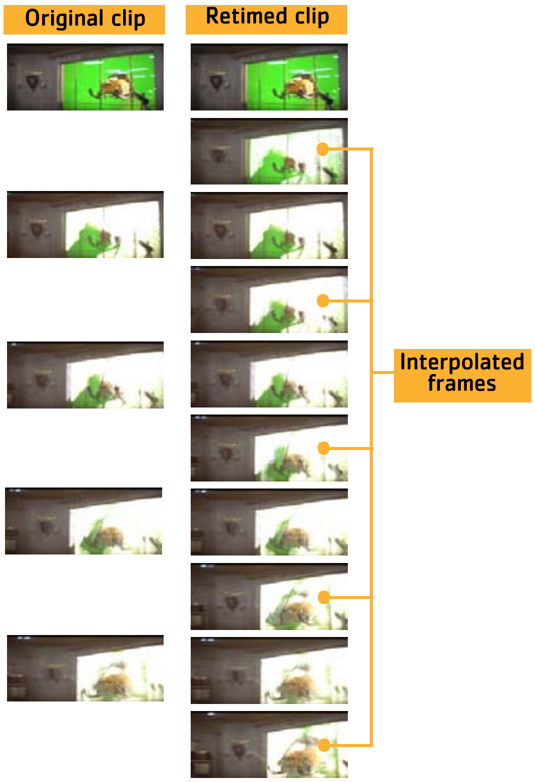


Figure 11.1: Interpolating frames.

The simplest way for Nuke to interpolate is to duplicate existing frames and increase the length of the clip—this is the default method of interpolation. However, this method can create jittery playback, especially when the image depicts very fast motion and the clip is retimed to be considerably longer than its original length. For such cases, Nuke provides different nodes for smoothly interpolating the frames.

Frame-Blending

The Frame Blend node interpolates frames by generating an additive composite of the frames that precede and follow it, rather than creating mere copies between the existing frames.

Here is an example of the difference between frame-copy and frame-blend interpolation. In the first frame of Figure 11.2, you see a copy of the preceding frame. In the second frame, you see a new image generated by blending the previous and subsequent frames.

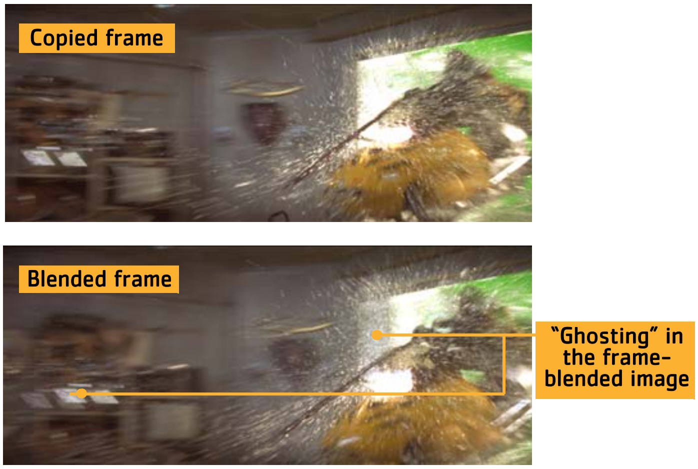


Figure 11.2: Frame-copied versus frame-blended interpolation.

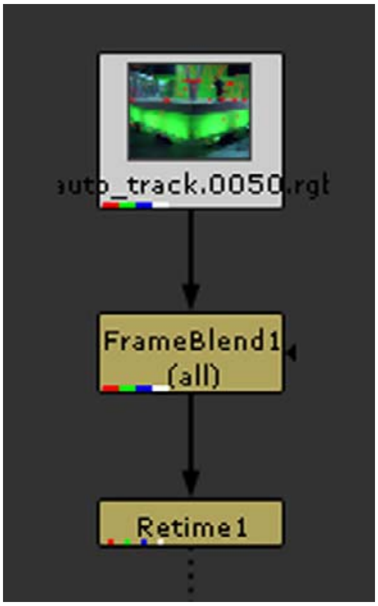
The latter method creates "ghosting" around all fast moving features (the window frame and the pages on the desk, for example). This may look odd when viewed as part of a still frame, but will contribute to smoother motion during actual playback.

You can enable frame-blending by manipulating the **shutter** value of a retiming node. Higher shutter values generate more frame-blending. Or, you can insert a Frame Blend node before the temporal effect you want to influence. The below figure shows an example of Frame Blend with the Retime node.

To insert a FrameBlend node:

1. Choose **Time > FrameBlend** from the menu.

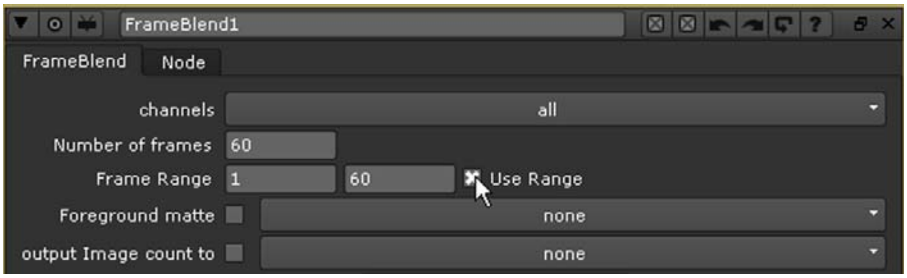
   Remember to place it upstream from the temporal effect you want to influence.



2. Enter the number of frames to blend in the **Number of frames** field.

   or

   Check the box for **Use Range** and enter the starting and ending frames that you want to blend



channels: all
Number of frames: 60
Frame Range: 1 60 ☒ Use Range
Foreground matte: ☐ none
output Image count to: ☐ none

3. If necessary, check **Foreground matte** and select the channel to limit the blending effect.

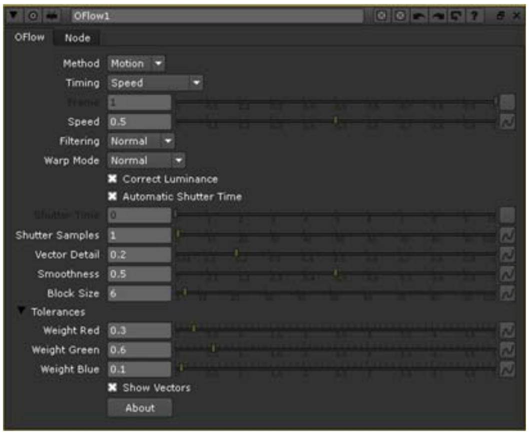
The option beneath Foreground matte (**output Image count to**) saves a floating point alpha image to a channel you specify; the result indicates the number of images that contributed to each pixel of the matte. To normalize the alpha, divide the number 1 by the number of frames averaged, and then multiply the alpha channel by this result. You can also use the inverse of this matte for additional degraining.

OFlow Retiming

The OFlow node generates high-quality retiming operations analyzing the movement of all pixels in the frames and then rendering new "in-between" images based on that analysis. This node can also add motion blur or enhance the existing motion blur in the image.

To retime with OFlow:

1. Select the node for the clip that you want to retime.
2. Choose **Time** > **OFlow** from the menu bar.
3. Set the speed of the output clip. A value of 0.5 will slow the movement down.



Method Motion
Timing Speed
Frame 1
Speed 0.5
Filtering Normal
Warp Mode Normal
☒ Correct Luminance
☒ Automatic Shutter Time
Shutter Time 0
Shutter Samples 1
Vector Detail 0.2
Smoothness 0.5
Block Size 6
Tolerances
Weight Red 0.3
Weight Green 0.6
Weight Blue 0.1
☒ Show Vectors

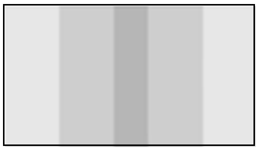
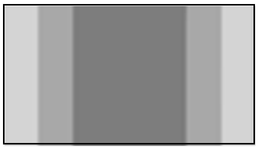
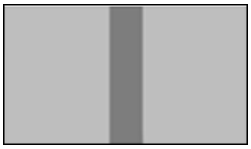
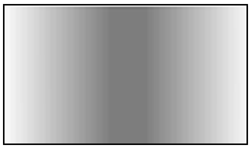
That's pretty much it.

If you prefer you can map input to output frames to retime the clip. For example, if you wanted to halve the speed of a 50 frame clip using this method, switch Timing to Source Frame. On frame 1 set a key for the Frame value to be 1. On frame 50 set a key for the Frame value to be 25.

OFlow Parameters

The following table describes the different parameters in the OFlow node's controls.

OFlow Parameter	Function
Method	Sets the interpolation algorithm. • **Frame** - the nearest original frame is displayed. • **Blend** - a mix between two frames is used for the in-between frame. This is quick to render and is useful when tweaking the timing on a curve before setting the method to motion. • **Motion** - vector interpolation is used to calculate the in-between frame.
Timing	Sets how to control the new timing of the clip. • **Speed** - select this if you wish to describe the retiming in terms of "double speed" or "half speed". • **Source Frame** - select this if you wish to describe the retiming in terms of "at frame 100 in the output clip I want to see frame 50 of the source clip". You'll need to set at least 2 key frames for this method to work.
Frame	This parameter is active only if timing is set to Frame. Use this to specify the source frame at the current frame in the time bar. For example, to slow down a 50 frame clip by half set the Frame to 1 at frame 1 and the Frame to 50 at frame 100. The resulting animation curve will result in a half-speed retime.
Speed	This parameter is only active if Timing is set to Speed. Values below 1 slow down the clip. Values above 1 speed up movement. For example, to slow down the clip by a factor of 2 (half speed) set this value to 0.5. Quarter speed would be 0.25.
Filtering	Sets the quality of the filtering when producing in-between frames. • **Normal** - uses bilinear interpolation which gives good results and is a lot quicker than extreme. • **Extreme** - uses a sinc interpolation filter to give a sharper picture but takes a lot longer to render.
Warp Mode	Sets how to control the new timing of the clip. • **Simple** - this is the quickest option, but may produce poor results around moving objects and image edges. • **Normal** - this is the default option with better treatment of moving objects and image edges. • **Occlusions** - this is the advanced option which attempts to reduce the level of background dragging that occurs between foreground and background objects.
Correct Luminance	Local motion estimation is highly dependent upon the idea that the brightness of objects don't vary through a sequence. Where brightness varies rapidly - for example a highlight moving across the bodywork of a car - the motion calculation will perform poorly. The luminance of a shot can come from other sources too - such as an overall flicker problem. In these cases where there is a global luminance shift, toggling this control on will allow the local motion estimation algorithm to take account of overall brightness changes between frames.

OFlow Parameter	Function
Automatic Shutter Time	Calculates the shutter time throughout the sequence automatically.
Shutter Time	Sets the equivalent Shutter Time of the retimed sequence. A shutter time of 1 is equivalent to averaging over plus and minus half an input frame which is equivalent to a shutter angle of 360 degrees. A shutter time of 0.5 is equivalent to a shutter angle of 180 degrees. Imagine a gray rectangle moving left to right horizontally across the screen. The figures below show how Shutter Time affects the retimed rectangle.  Shutter Time 1  Shutter Time 0.5
Shutter Samples	Sets the number of in-between images used to create an output image during the shutter time. Increase this value for smoother motion blur, but note that it takes much longer to render.  Shutter Samples 2  Shutter Samples 20
Vector Detail	Adjust this to vary the resolution of the vector field. Large vector fields take longer to process, but contain more detail and may help to isolate smaller motion in the scene. A value of 1 will generate a vector for every pixel. A value of 0.5 will generate a vector at every other pixel. For some sequences, a high vector detail near 1.0 generates too much unwanted local motion detail and often a low value will give a better result.
Smoothness	Vector fields usually have two important qualities: they should accurately match similar pixels in one image to another and they should be smooth rather than noisy. Often it is necessary to trade one of these qualities off against the other. A high **smoothness** will miss lots of local detail, but is less likely to provide you with the odd spurious vector. A low **smoothness** will concentrate on detail matching, even if the resulting field is jagged. The default value of 0.5 should work well for most sequences.

OFlow Parameter	Function
Block Size	The vector generation algorithm subdivides the image into small blocks, and separately tracks them. **blockSize** defines the width and height of these subdivisions. Smaller values will produce noisy data, whereas larger values may produce data that is lacking in detail. This value should rarely need editing; some sequences may benefit from using large block sizes to help the algorithm track regions better where the algorithm isn't "locking on" to the overall motion in the sequence.
Tolerances	For efficiency, much of the local motion estimation is done on luminance only - i.e. using monochrome images. The tolerances parameters allow you to tune the weight of each color channel when calculating the image luminance. These parameters rarely need tuning. However, you may, for example, wish to increase the red weighting **Weight Red** to allow the algorithm to concentrate on getting the motion of a primarily red object correct, at the cost of the rest of the items in a shot.
Weight Red	The red weighting used when calculating the vector field.
Weight Green	The green weighting used when calculating the vector field.
Weight Blue	The blue weighting used when calculating the vector field.
Show Vectors	Switch this on to draw the motion vectors over the image.
About	Shows the version number of this node.

Tip *Old-time Nuke users may remember nodes called OpticalFlow and OptiFlow. OFlow replaces these nodes. It can be used for retiming and adding motion blur, but it does not have the capability to output motion vectors. To output motion vectors, you could use F_VectorGenerator (included in The Foundry's Furnace plug-ins).*

Warping Clips

Warping refers to slowing down, speeding up, or even reversing select frames in a clip without necessarily altering its overall length. Otherwise stated, warps add, subtract, or reverse the temporal detail in a range of frames within a clip. For example, Figure 11.3 depicts a snowmobile clip (downsampled to just ten frames for easy representation) that we might want to warp.

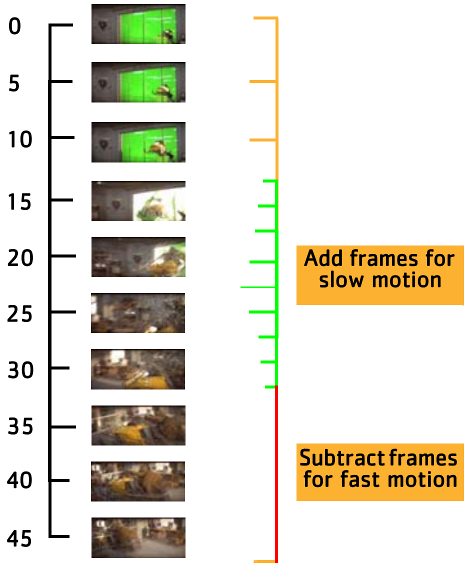

0
5
10
15
20
25
30
35
40
45
Add frames for slow motion
Subtract frames for fast motion

Figure 11.3: Planning the time warp.

One way—in fact, kind of the classic way—to warp this clip would be to play the frames just prior to the collision at their original rate, the frames involving the collision in slow motion, and the frames after the collision in fast motion.

You could achieve such a warp by sculpting the curve in Nuke's TimeWarp curve, which is a part of the Retime node's parameters, to look something like the one depicted below.

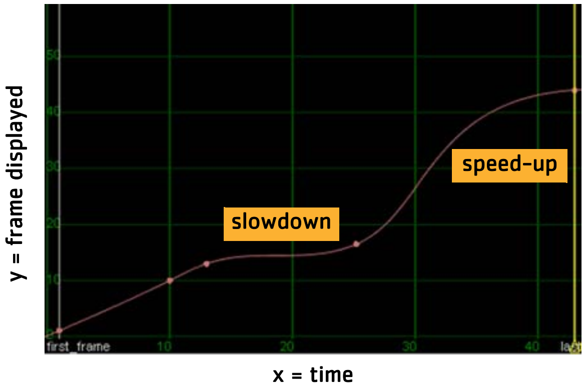


Figure 11.4: Editing the warp curve.

The basic "rules" for editing the warp curve are as follows:

- To slow down motion, decrease the slope of the curve.

- To speed up motion, increase the slope of the curve.
- To reverse motion, create a downward sloping portion on the curve (a dip, in other words).

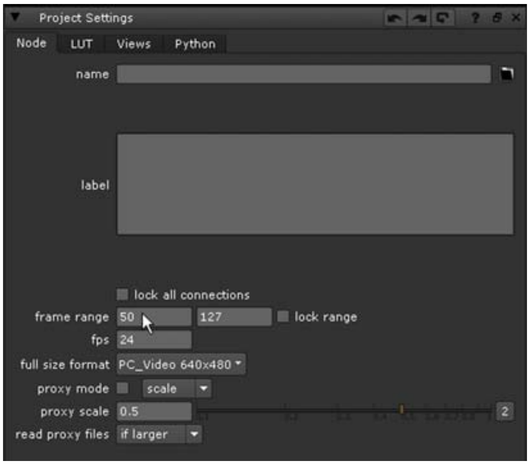
To warp a clip:

1. Click **Time > Retime** to insert a Retime node into your script.
2. Click the **TimeWarp** tab to reveal the TimeWarp curve.
3. Attach a Viewer to this node, so you can see the effect of your changes.
4. Sculpt the TimeWarp curve according to the rules above. (**Ctrl/Cmd+Alt** click to insert keyframe knots on the curve; **Ctrl/Cmd**+drag to reposition keyframe knots; **Ctrl/Cmd**+drag to rotate a keyframe knot control handles.)
5. If you want to enable frame blending on the output, either input a value larger than one in the Retime node's **shutter** parameter, or insert a FrameBlend node prior to the Retime node.

Global Frame Range and Speed

Nuke automatically adjusts the timeline of every Viewer window you open to show the "in" and "out" frames for the clip you're viewing.

After you retime a clip in your compositing script, you may need to adjust the script's global frame range and playback speed (frames-per-second), to account for the retiming operations.



☐ lock all connections
frame range 50 127 ☐ lock range
fps 24
full size format PC_Video 640x480
proxy mode ☐ scale
proxy scale 0.5 2
read proxy files if larger

Figure 11.5: Adjusting the global frame range.

Choose **Edit > Project settings** (or press **S** over the Nuke window) and then enter the new **frame range** and **fps** in the Project settings properties panel.

Applying the TimeBlur Filter

When a fast moving subject is recorded on film or video, its edges appear to smear as a result of the object's movement while the shutter is open. The longer the shutter remains open at each frame interval, the more obvious this effect. TimeBlur simulates this phenomenon by sampling its input at "divisions" times over "shutter" frames starting at the current frame plus "offset".

Time blur is commonly applied to garbage masks that are tracked to a fast moving feature. The time blur averages the incoming mask image over the shutter period, to better match the motion blur in the original image and creating a more convincing integration.

To apply motion blur to a clip:

1. Click **Time** > **TimeBlur** to insert a TimeBlur node into your script. Place it downstream from the element to which you want to apply motion blur.
2. In the TimeBlur properties panel, set **divisions** to the number of times you want to sample the input over the shutter time. For images with fast-moving content higher values will be necessary to eliminate "steppiness" or banding in the output.
3. Set **shutter** to equal the span of time (in frames) over which the input should be sampled. A shutter time of .5 is typical and would correspond with a camera shutter of 180 degrees.
4. Set the **shutteroffset** to control when the sampling of the input starts relative to the frame being rendered, analogous to when the camera shutter opened to capture corresponding film or video footage you might have at the same frame. You may need to adjust this by eye to align, for example, a garbage mask with an underlying feature.

Tip *You may find that using TimeBlur on all the upstream nodes in your composition can be unnecessary and very time consuming. In these cases, you can use NoTimeBlur node to limit the number of nodes to which you're applying TimeBlur. Just insert the NoTimeBlur node in your node tree above the TimeBlur and any nodes you want the TimeBlur node to process.*

Editing Clips

As a node-based system, Nuke doesn't have a timeline. Nevertheless, you can still perform editorial operations that you might associate with a timeline. You can slip clips (move them forward or backward in time), cut them, or splice them to other clips.

Slipping Clips

Slipping a clip refers to moving it backward or forward in time. There are any number of reasons why you might want to do this (for example, to synchronize events in a background and foreground clip).

To slip a clip:

1. Click **Time > TimeOffset** to insert a TimeOffset node into your script. (Place it downstream from the element to which you want to slip.)
2. Attach a Viewer to this node, so you can see the effect of your changes.
3. In the TimeOffset properties panel, check **reverse input** if you want to invert the clip (make the last frame the first, and so on).
4. In the **time offset (frames)** field, type the number of frames by which you want to slip the clip. Enter a negative value to subtract frames from the head of the clip. Enter a positive value to add frames to the head of the clip.
5. Adjust the script length for the new output range. Choose **Edit** > **Project settings**, and enter **frame range** values that match the output range you specified.

Note *It's not mandatory that you adjust the script's frame range after slipping the clip. If you don't, the Viewer will fill the empty frames at the tail of the clip by holding on the last frame.*

Cutting Clips

Cutting a clip refers to shortening it by removing frames from its head or tail.

To cut a clip:

1. Click **Time** > **FrameRange** to insert a FrameRange node into your script. Insert it downstream from the element to which you want to cut.
2. Attach a Viewer to this node, so you can see the effect of your changes.
3. In the **frame range** fields, then enter the appropriate in and out point values.

   For example, if your original clip is 50 frames but you want to use only the last 25 frames in your composite, you would enter **25** as the In point and leave the Out point at **50**.
4. Adjust the script length for the new output range. Choose **Edit** > **Project settings**, and enter **frame range** values that match the output range you specified.

Note *It's not mandatory that you adjust the script's frame range after cutting the clip. If you don't, the Viewer will simply fill the empty frames at the head or tail of the clip by holding on the first or last frame.*

Splicing Clips

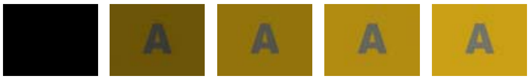
Splicing refers to joining clips head-to-tail, thus allowing action to flow from one shot to the next. When you splice clips, you have options for:

- Fading to or from black.
- Dissolving from the first to second clip.
- Slipping the combined clip in time.

To splice clips:

1. Click **Time** > **AppendClip** to insert an AppendClip node into your script.
2. Attach its **1** and **2** pipes to the clips you want to join. (The clip attached to pipe 1 will precede the one attached to pipe 2.)
3. Attach a Viewer to this node, so you can see the effect of your changes.
4. If necessary, expand the script length to accommodate the total length of the newly merged clip:
   - Click **Edit > Project settings**. The Project settings properties panel appears.
   - Enter **frame range** values that matches the total length.
5. In the **Fade In** and **Fade Out** fields of the AppendClip properties panel, type the number of frames, if any, you want to fade to or from black.

   For example, typing a **5** in the **Fade In** field would result in the following effect at the head of the merged clip.

   (The inverse of this effect would occur at the *tail* of the merged clip were you type **5** in the **Fade Out** field.)
6. In the **Cross Dissolve** field, type the number of frames, if any, of overlap you want between the first and second clip.

   For example, leaving **Cross Dissolve** at the default of **0** creates a simple cut—the transition from the first to second clip is instantaneous. Typing in a **5** creates a time span of five frames in which the first clip's gain ramps downward to zero, while the second's ramps upward to 100%.

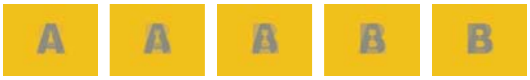


Figure 11.6: Dissolve.

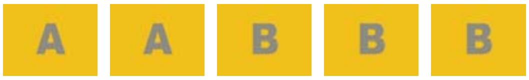


Figure 11.7: Cut.

7. In the **First Frame** field, type the number of frames, if any, by which you want to slip the clip. Enter a negative value to subtract frames from the head of the merged clip. Enter a positive value to add frames to the head of the clip.
8. Slipping the merged clips may create empty black frames at its head or tail. As appropriate, choose **First frame** or **Last frame** if you want these empty frames to appear as copies of the first or last frame.

12 WARPING AND MORPHING IMAGES

Nuke's warping and morphing tools allow you to distort elements in an image, apply and correct lens distortions, and morph one image into another. The nodes designed for these operations include the GridWarp node, the SplineWarp node, and the iDistort node. In this chapter, we focus on the GridWarp and SplineWarp nodes.

Warping

Warping refers to manipulating an image so that elements in the image are distorted. Unlike many of the transformations described under "Transforming Elements" on page 226, warps are transformations that only affect some of the pixels in an image rather than all of them. For example, you might make an animal's eyes bigger or a person's smile wider without affecting the rest of their features.

This is not to say that the pixels around the area you are moving do not move with the area. They do, because accommodating the change this way often produces more realistic results. However, the distortion lessens the further you get from the moved pixels. You also have some control over which pixels are moved and which are not, and can isolate the warp to a small area. Still, in an ideal situation, the subject you are going to warp is a subject you can key out or rotoscope to isolate it from its background before you create the warp. This way, you can be sure that the background stays intact.

In addition to performing creative manipulations on the shapes of the subjects in your images, you can also use warping to simulate different types of film or video lenses or to remove unwanted lens distortions.

Below, we discuss how to warp images, first using the GridWarp node and then the SplineWarp node. Finally, we also teach you to animate the warps. Again, we start with the GridWarp node and then show you how to do the same with the SplineWarp node.

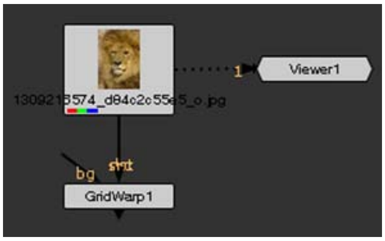
Warping Images Using the GridWarp Node

The GridWarp node allows you to warp images by transferring image information from one Bezier or B-spline grid onto another. When using this node, you first create the source grid, which defines where to warp from. Next, you create the destination grid, which defines where to warp the image to. This grid can be a duplicate of the source grid, or you can draw it separately. When you manipulate the destination grid, the corresponding warp is applied to the source image.

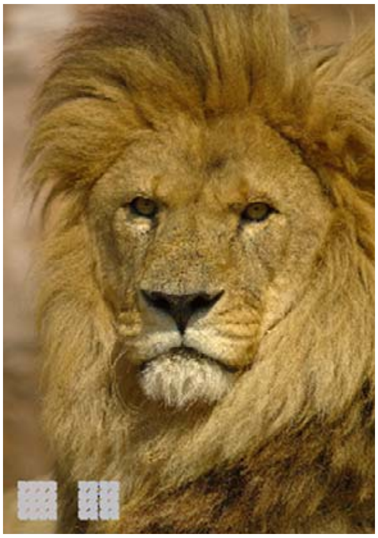
The GridWarp node also includes controls for animating the warp, adding motion blur to warps that occur quickly, and selecting the level of antialiasing used to remove any artifacts the warp may have caused.

To warp an image using the GridWarp node:

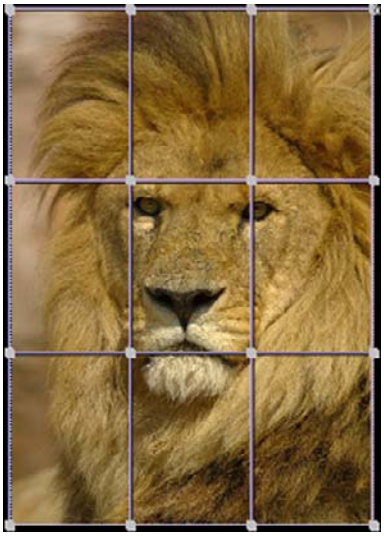
1. Select **Transform** > **GridWarp** to insert a GridWarp node after the image you want to warp.
2. Connect both the **src** and the **dst** input and a Viewer to the image.

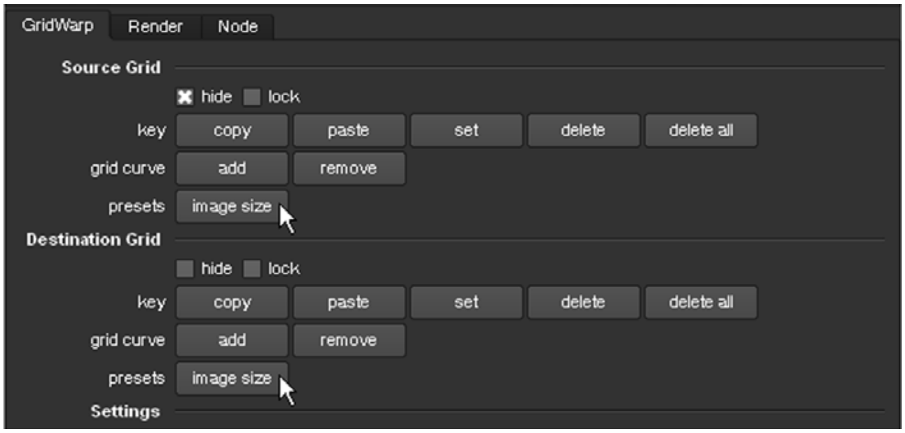


3. When the GridWarp properties panel is open, you can see the source and destination grids appear as small overlays in the Viewer. The source grid is pink, and the destination grid blue. In the following steps, you use the pink source grid to define which areas you want to warp and the blue destination grid to define where to warp these areas to.



4. To make the grids the same size as the input image, click the **image size** buttons under both **Source Grid presets** and **Destination Grid presets.**





GridWarp | Render | Node

Source Grid

☒ hide ☐ lock

key: copy | paste | set | delete | delete all

grid curve: add | remove

presets: image size

Destination Grid

☐ hide ☐ lock

key: copy | paste | set | delete | delete all

grid curve: add | remove

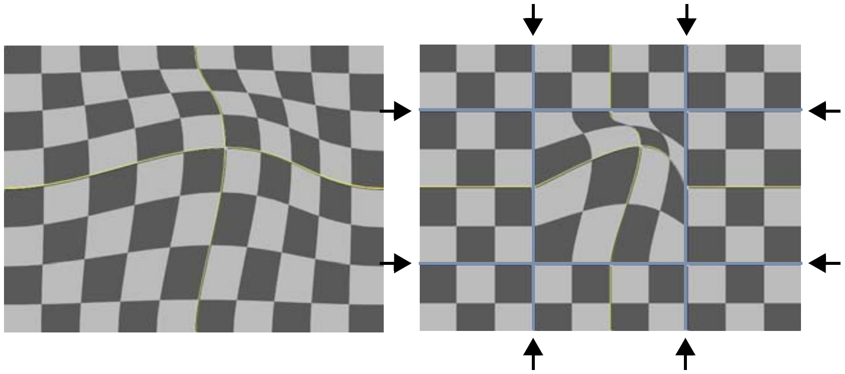
presets: image size

Settings

5. For now, check **hide** under **Destination Grid** to hide the blue destination grid in the Viewer. This way, you can't accidentally distort the image yet.
6. Modify the grid around the area you want to warp. Usually, you want the grid to conform to the subject of the source image. For example, if you are warping an animal's eyes, you need to create grid lines that follow the edges of the eyes.

   To add more points and lines to the grid, click the **add** button under the **Source Grid** controls and click on an existing grid line in the Viewer. If you click on a horizontal line, a vertical line is added to the grid. If you click on a vertical line, a horizontal line is added to the grid. The lines can be further apart in the areas that you don't intend to warp.

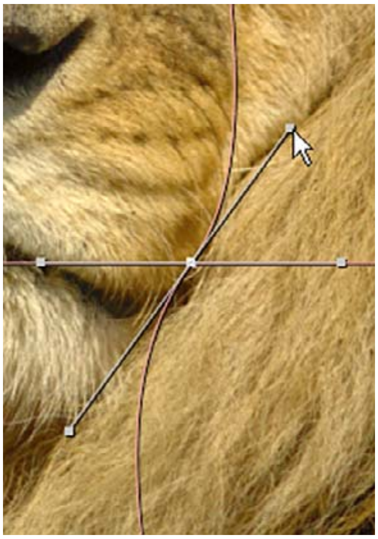
   You can use the grid lines to isolate the areas you do not want to warp. You do this by adding lines between the area you intend to warp and the area you don't want to change.



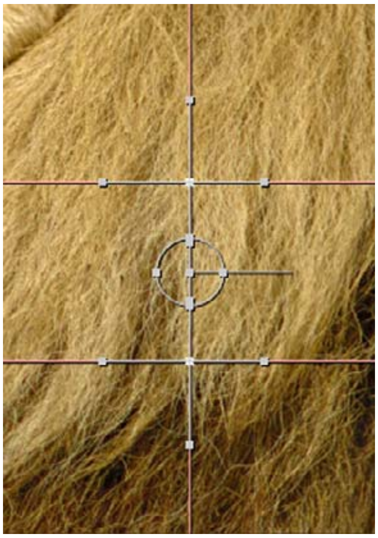
An unlimited warp

The same warp limited to a small area with grid lines

When you select a point, four tangent handles appear around it. You can use these handles to modify the curves connecting the points.

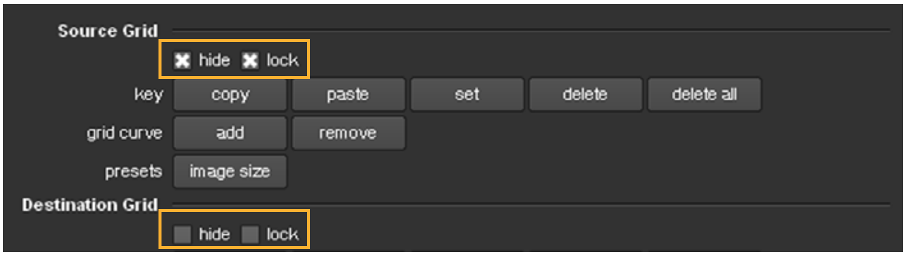


To move several points together, draw a marquee around them and use the transformation overlay that appears. For more information on how to use the transformation overlay, see "Using the 2D Transformation Overlay" on page 226.



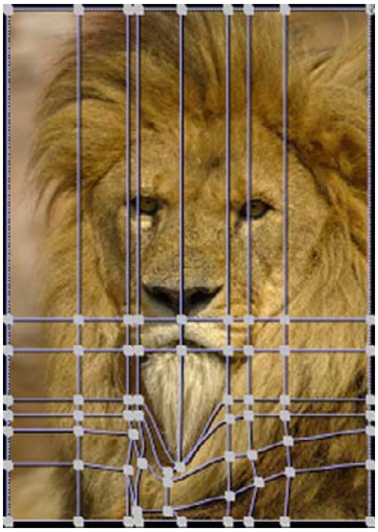
To remove a line, click the **remove** button and the line you want to remove.

7. If necessary, animate the source grid to match any movement in the source image. For more information on how to do this, see "Animating Warps" on page 398.
8. Unless you want to draw the destination grid separately, click **copy** under **Source Grid** and **paste** under **Destination Grid**. This copies the source grid you created in the previous step into the destination grid.
9. Connect the Viewer to the GridWarp node.
10. Hide the source grid (check **hide**) and lock it to prevent accidental changes (check **lock**). Show the destination grid instead (uncheck **hide**). It should look the same as the source grid, only blue (assuming you copied the source grid into the destination grid).

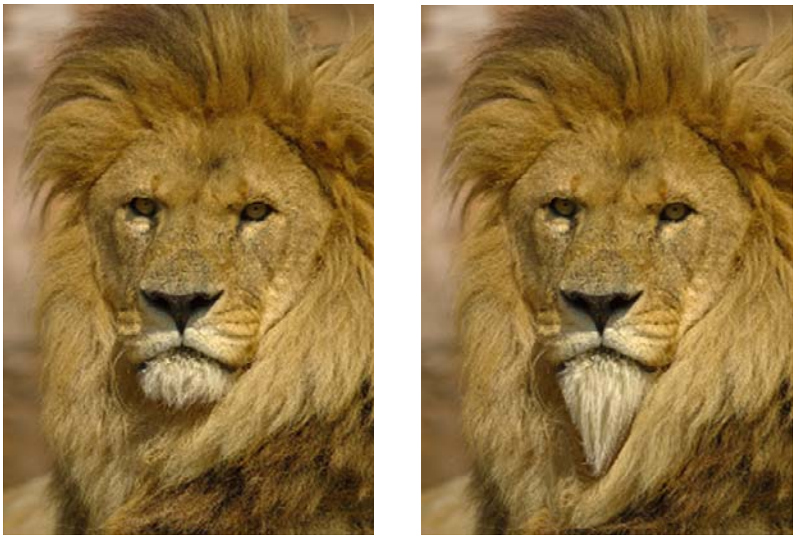

Source Grid
☒ hide ☒ lock
key
copy
paste
set
delete
delete all
grid curve
add
remove
presets
image size
Destination Grid
☐ hide ☐ lock

11. In the areas where you want to warp the image, drag the points on the grid to a new position. The pixels in these areas are moved in the direction you moved the points. Pixels in the nearby areas are also moved to accommodate the change, but the distortion lessens the further you get from the repositioned points. If you don't want a nearby

area distorted, add more grid lines between the area and the points you want to move before you drag the points to a new location.



12. To better see what the warped image looks like, press **O** on the Viewer to toggle the overlay off. To compare the original and warped images, press **D** repeatedly on the GridWarp node to disable and enable it. If you see changes in the areas you don't want to warp, go back to modifying the destination grid.



13. If necessary, animate the destination grid to match any movement in the destination image.

14. If necessary, adjust the controls described in the following table.

Control	What it does
reverse	Check this to invert the distortion.
submesh resolution	Set the number of subdivisions that are created between Bezier or B-spline curves in the grid. The higher the value, the more accurate the distortion between the grid lines.
distortion	The overall distortion is multiplied by this amount. The lower the value, the less the image is distorted. At 0, you get the source image, while at 1 you get the destination image. At 0.5, the distortion is halfway between the source and the destination images.
blend	Dissolve between the source image (at 0) and the destination image (at 1).
background	The warped image is rendered on top of an unwarped background. This control sets what to use as that background: • **on black** - Render the warped image on top of a constant black image. • **on src** - Render the warped image on top of the image connected to the **src** input of the GridWarp node. • **on dst** - Render the warped image on top of the image connected to the **dst** input of the GridWarp node. • **on bg** - Render the warped image on top of a background image connected to the **bg** input of the GridWarp node.
background blend	Blend between the output of the GridWarp node (at 0) and whatever you have selected from the **background** pulldown menu (at 1).
filter (on the **Render** tab)	Choose the appropriate filtering algorithm (see "Choosing a Filtering Algorithm" on page 227).
antialiasing (on the **Render** tab)	Select the level of antialiasing to reduce any aliasing artifacts the warp may have caused.
samples (on the **Render** tab)	Enter the number of samples to render per pixel when motion blurring. The higher the number, the smoother the result. Setting the value to 0 produces no motion blur.
shutter (on the **Render** tab)	Enter the number of frames the shutter stays open when motion blurring. For example, a value of 0.5 would correspond to half a frame. Increasing the value produces more blur, and decreasing the value less.

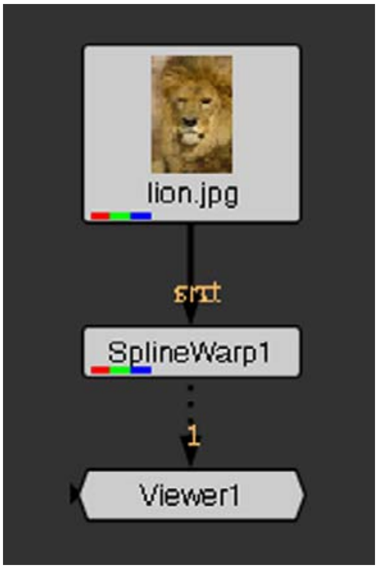
Control	What it does
shutteroffset (on the **Render** tab)	Select when the shutter opens and closes in relation to the current frame value when motion blurring: • to center the shutter around the current frame, select **centred**. For example, if you set the **shutter** value to 1 and your current frame is 30, the shutter will stay open from frame 29,5 to 30,5. • to open the shutter at the current frame, select **start**. For example, if you set the **shutter** value to 1 and your current frame is 30, the shutter will stay open from frame 30 to 31. • to close the shutter at the current frame, select **end**. For example, if you set the **shutter** value to 1 and your current frame is 30, the shutter will stay open from frame 29 to 30. • to open the shutter at the time you specify, select **custom**. In the field next to the pulldown menu, enter a value (in frames) you want to add to the current frame. To open the shutter before the current frame, enter a negative value. For example, a value of - 0.5 would open the shutter half a frame before the current frame.
randomize time (on the **Render** tab)	Adjust this to add randomness to the distribution of samples in time so they don't produce regularly spaced images. The larger the value, the larger the time difference between the samples.
sample diameter (on the **Render** tab)	Adjust the diameter of the circle that the samples for each pixel are placed in for antialiasing. The larger the value, the more pixels are jittered.

Warping an Image Using the SplineWarp Node

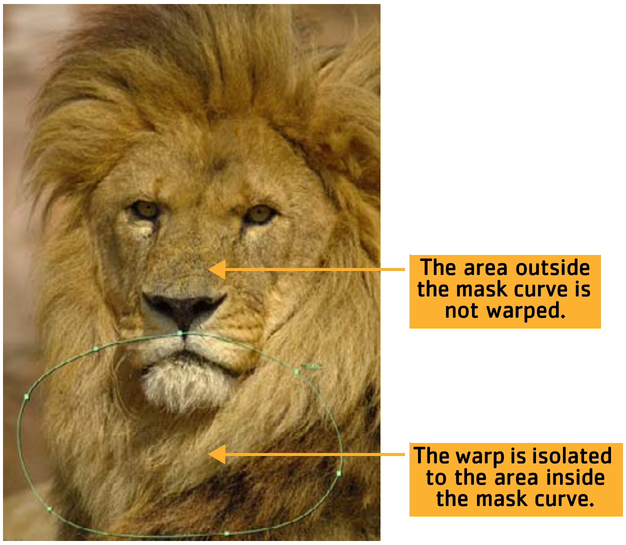
The SplineWarp node deforms an image based on multiple Bezier or B-spline curves that you create. The source curve defines where to warp from, while the destination curve defines where to warp the source image to. Unlike with the GridWarp node, you can draw these curves anywhere on the image rather than only add points on the existing grid lines. The controls for adding and modifying points are similar to the controls of the RotoPaint node.

To warp an image using the SplineWarp node:

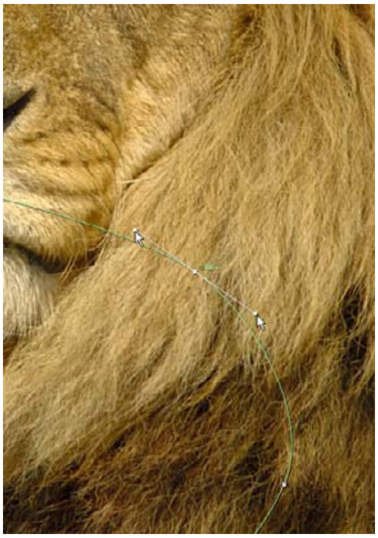
1. Select **Transform** > **SplineWarp** to insert a SplineWarp node after the image you want to warp.
2. Connect both the **src** and the **dst** input to the image. Attach a Viewer to the SplineWarp node.


lion.jpg
src
SplineWarp1
1
Viewer1

3. If you need to animate the warp, make sure **autokey** is checked in the SplineWarp node's controls. This way, the curves you will soon create will automatically be set as key shapes for the animation.
4. Make sure **show both curves** is not checked and **show** is set to **blend**.
5. If there are areas in the image that you do not want to warp, check **first bezier masks deformation**. Then, Ctrl+Alt+click (Mac users Cmd+Alt+click) on the image in the Viewer to draw a curve that isolates the area you intend to warp from the area you want to keep untouched.


The area outside the mask curve is not warped.
The warp is isolated to the area inside the mask curve.

When you select a point, two tangent handles appear around it. You can use these handles to modify the curves connecting the points.



To move several points together, draw a marquee around them and use the transformation overlay that appears. For more information on how to use the transformation overlay, see "Using the 2D Transformation Overlay" on page 226.

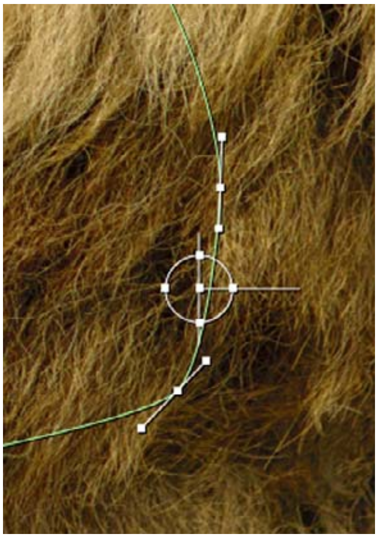
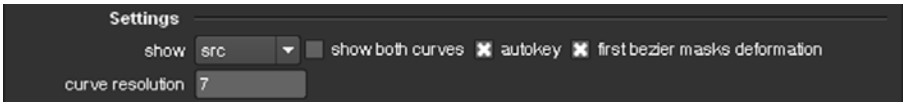


Figure 12.1: To close an open curve or open a closed curve, right-click on a point and select **open/close curve**.

To remove a point, right-click on the point and select **delete point**.

To remove an entire curve, right-click on a point on the curve and select **delete curve**.

6. Set **show** to **src** so you can create the source curves that define where to warp from.



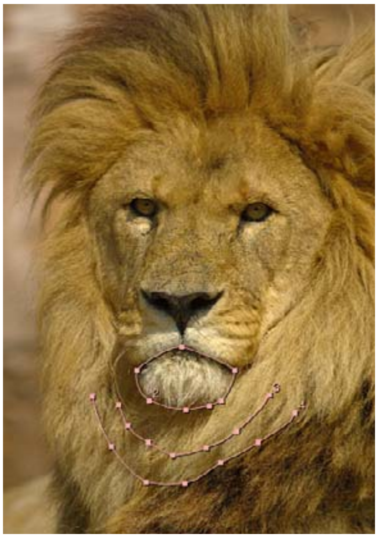
Settings

show src ☐ show both curves ☒ autokey ☒ first bezier masks deformation

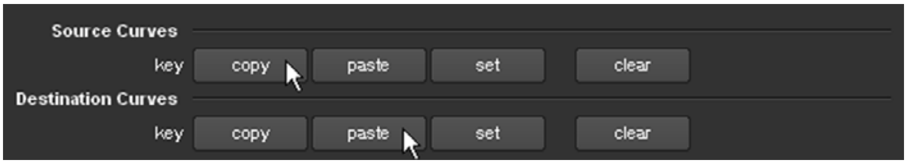
curve resolution 7

7. In the Viewer, **Ctrl+Alt+click** (Mac users **Cmd+Alt+click**) to create the pink source curves. Usually, you want the curves to conform to the subject of the source image. For example, if you are warping an animal's eyes, you need to create curves that follow the edges of the eyes.

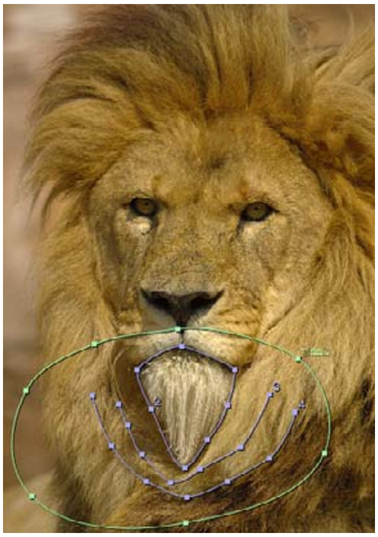
   You probably want to create more than one curve around the area you are warping. To begin a new curve, make sure none of the existing points are selected and **Ctrl/Cmd+Alt+click** on the image. The curves are numbered in the order you create them.



8. If you need to animate the warp, scrub to another frame on the timeline and adjust the curves to match any movement in the source image. If **autokey** is checked, key shapes are set automatically. For more information on how to animate the warp, see "Animating Warps" on page 398.
9. Unless you want to create the destination curves separately, click **copy** under **Source Curves** and **paste** under **Destination Curves** to use the curves you just created as the basis for the destination curves.



10. Then, set **show** to **srcwarp**. You should now see the blue destination curves that are identical to the source curves (assuming you copied the source curves into destination curves).
11. In the areas where you want to warp the image, drag the points on the curves to a new position. The pixels in these areas are moved in the direction you moved the points.



12. To better see what the warped image looks like, press **O** on the Viewer to toggle the overlay off. To compare the original and warped images, press **D** repeatedly on the SplineWarp node to disable and enable it. If you see changes in the areas you don't want to warp, go back to modifying the mask or the destination curves.
13. If necessary, move to other frames and animate the destination grid to match any movement in the destination image.
14. If necessary, adjust the controls described in the following table.

Control	What it does
show	Select the image to output: - **src** - the unwarped image from the **src** input. - **srcwarp** - the warped image from the **src** input. - **dst** - the unwarped image from the **dst** input. - **dstwarp** - the warped image from the **dst** input. - **blend** - a dissolve between the warped images from the **src** and **dst** inputs.
show both curves	Check this if you want to display both the source and destination curves in the Viewer at the same time.

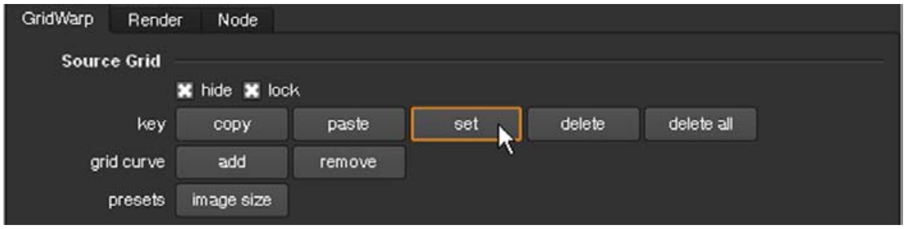
Control	What it does
curve resolution	Increase this value to improve the accuracy of how the warp matches the curves.
preview resolution	Increase this value to improve the quality of the OpenGL preview.
distortion	The overall distortion is multiplied by this amount. The lower the value, the less the image is distorted. At 0, you get the source image, while at 1 you get the destination image. At 0.5, the distortion is halfway between the source and the destination images. To see the effect, you need to set **show** to **blend**.
blend	Dissolve between the source image (at 0) and the destination image (at 1). To see the effect, you need to set **show** to **blend**.

Animating Warps

Unless you are warping a still image, you probably want to animate the warp. In the GridWarp and SplineWarp node's properties panels, there are controls for animating both the source and the destination grids or curves. Here, we first look at the GridWarp node and then the SplineWarp node. The instructions assume you know how to warp a still image using these nodes (if not, refer to "Warping Images Using the GridWarp Node" on page 386 and "Warping an Image Using the SplineWarp Node" on page 393).

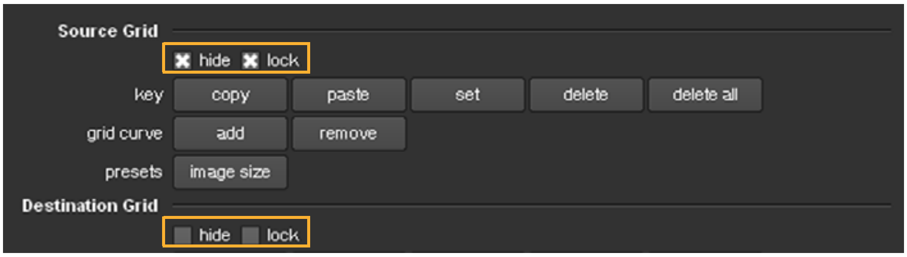
To animate a warp using the GridWarp node:

1. While viewing the source image, adjust the pink source grid as necessary (see "To warp an image using the GridWarp node:" on page 387 for information on how to do this). When you are happy with the grid, click the **set** button under **Source Grid**. This saves the current grid as a key shape.



2. Move to a new frame and adjust the source grid accordingly. A new key shape is set automatically.
3. Repeat the previous step as necessary. If you need to delete a key shape, scrub to the frame where you set it and click **delete** under **Source Grid**. If you want to delete the entire animation and make the current grid the source grid for all frames, click **delete all**.

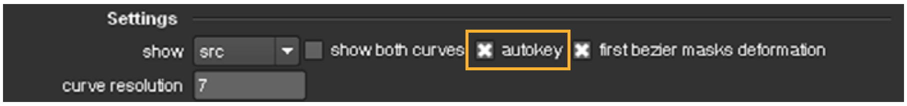
4. Click **copy** under **Source Grid** and **paste** under **Destination Grid** to copy the source grid into the destination grid.
5. Hide the pink source grid and display the blue destination grid.


Source Grid
☒ hide ☒ lock
key
copy
paste
set
delete
delete all
grid curve
add
remove
presets
image size
Destination Grid
☐ hide ☐ lock

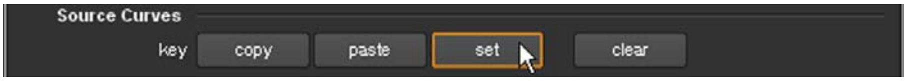
6. While viewing the output of the GridWarp node, adjust the destination grid until you are happy with the warp. Click the **set** button under **Destination Grid**. The current grid is saved as a key shape.
7. Move to a new frame and adjust the destination grid again. The modified grid is automatically set as a key shape.
8. Repeat the previous step until you are happy with the animated warp.

To animate a warp using the SplineWarp node:

1. If you want the key shapes created automatically when you create the curves, make sure **autokey** is checked in the SplineWarp controls.


Settings
show
src
☐ show both curves
☒ autokey
☒ first bezier masks deformation
curve resolution
7

2. Create the source curves as instructed under "To warp an image using the SplineWarp node:" on page 393. If **autokey** is checked, the current curves are saved as key shapes. If autokey is not checked, you need to click the **set** button under **Source Curves** to set the curves as key shapes.


Source Curves
key
copy
paste
set
clear

3. Scrub to a new frame on the timeline, and adjust the curves to conform to the movement of the image. If you have not checked autokey, click **set** under **Source Curves** again.
4. Repeat the previous step as necessary until you are happy with the source curve animation. If you need to delete a key shape, scrub to the frame where you set it and click **clear** under **Source Curves**. The

current key shape is deleted and the curves for the frame are calculated by interpolation.

5. Click **copy** under **Source Curves** and **paste** under **Destination Curves** to copy the source curves into the destination curves.
6. Animate the destination curves in the same way as the source curves (only using the controls under **Destination Curves**).

Morphing

Morphing refers to dissolving two images together so that the subject of one image seems to change shape and turn into the subject of the other through a seamless transition. A morph can be easily noticeable or very subtle. An example of a noticeable morph would be a man turning into a woman or one animal turning into another, whereas a transition from an actor to his stunt man would result in a much more subtle morph.

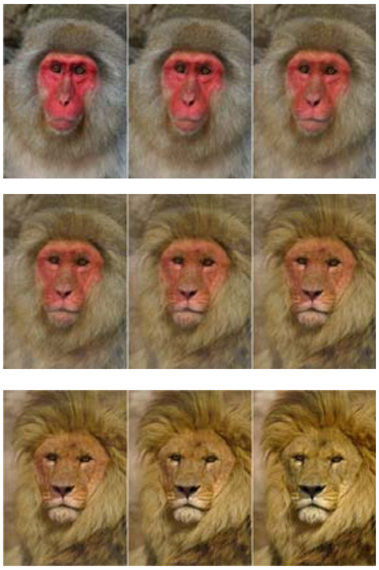


Figure 12.2: An image of a monkey turning into an image of a lion.

Morphing can be a time-consuming task, but it can be made easier by good advance planning of the shots. The more similar the characteristics, position, and movement of the subjects you want to morph are, the easier it is to morph them together. If the position and the movement of the subjects do not match, however, you can try to reposition and retime the clips before morphing them together. To do so, use the nodes described in Chapter 5, "Transforming Elements", on page 226 and Chapter 11, "Temporal Operations", on page 371. You can also use the Tracker node (see Chapter 6, "Tracking and Stabilizing", on page 246) to track the features you want to morph over time or to stabilize your clips before morphing them together.

Tip *You can copy animation data from the Tracker node to the points on the GridWarp/SplineWarp grid/curves. Do the following:*

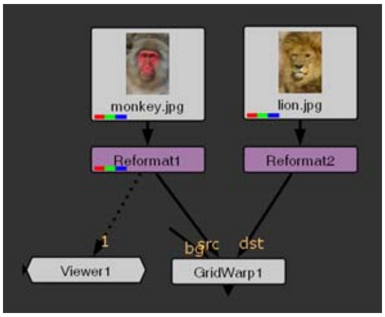
1. Make sure both the Tracker and the GridWarp/SplineWarp properties panels are open.

*2. **Ctrl/Cmd**+drag the animation button from the Tracker node properties on top of the individual grid/curve point in the Viewer. When you release the mouse, the point follows the animation from the Tracker node.*

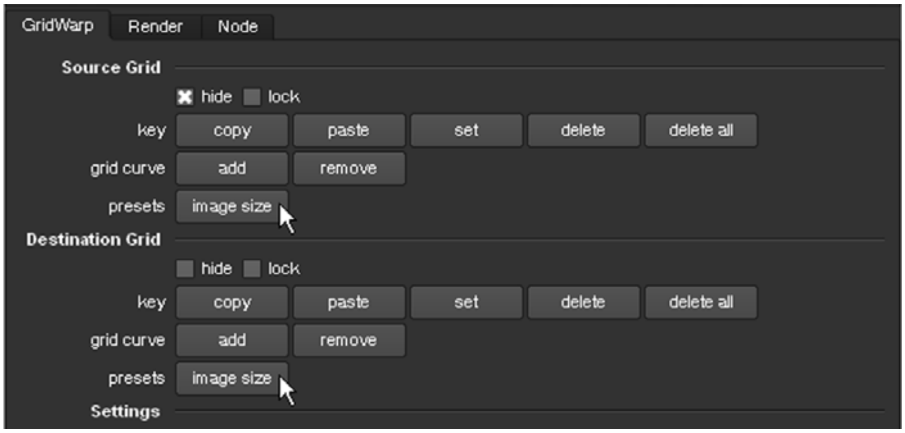
Below, we first discuss morphing images using the GridWarp node and then using the SplineWarp node.

To morph one image into another using the GridWarp node:

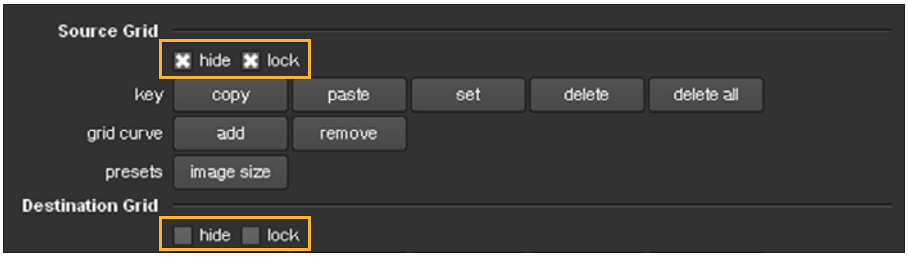
1. Select **Image** > **Read** to import the two images you want to morph together.
2. If the images do not have the same resolution, insert Reformat nodes after them. For more information, see "Reformatting Image Sequences" on page 139.
3. Select **Transform** > **GridWarp** to insert a GridWarp node into your script.
4. Connect the source image (the image you want to turn into another image) to the **src** input of the GridWarp node, and the destination image (the image you want the source image turned into) to the **dst** input. Connect a Viewer to the source image (or its Reformat node if one exists).



5. To make the grids the same size as the input images, click the **image size** buttons under both **Source Grid presets** and **Destination Grid presets**.


GridWarp
Render
Node
Source Grid
☒ hide ☐ lock
key
copy
paste
set
delete
delete all
grid curve
add
remove
presets
image size
Destination Grid
☐ hide ☐ lock
key
copy
paste
set
delete
delete all
grid curve
add
remove
presets
image size
Settings

6. In the GridWarp controls, check **hide** under **Destination Grid**.
7. Identify some key features in the source image that correspond to features in the destination image, and adjust the pink source grid to conform to these features. For more information on how to adjust the grid, see "To warp an image using the GridWarp node:" on page 387.
8. Click **copy** under **Source Grid** and **paste** under **Destination Grid**. This copies the source grid you created in the previous step into the destination grid.
9. Uncheck **hide** under **Destination Grid**. Instead, check **hide** and **lock** under **Source Grid**.


Source Grid
☒ hide ☒ lock
key
copy
paste
set
delete
delete all
grid curve
add
remove
presets
image size
Destination Grid
☐ hide ☐ lock

10. Attach the Viewer to the destination image or its Reformat node if one exists. You should now see the destination image and the blue destination grid.
11. Adjust the destination grid to conform to the subject of the destination image. This way, you warp the source image into the shape of the destination image.
12. Attach the Viewer to the GridWarp node and scrub to the frame where you want the morph to begin.
13. In the GridWarp controls, bring the **distortion** slider down to 0 (the source image). Click on the animation button and select **Set**

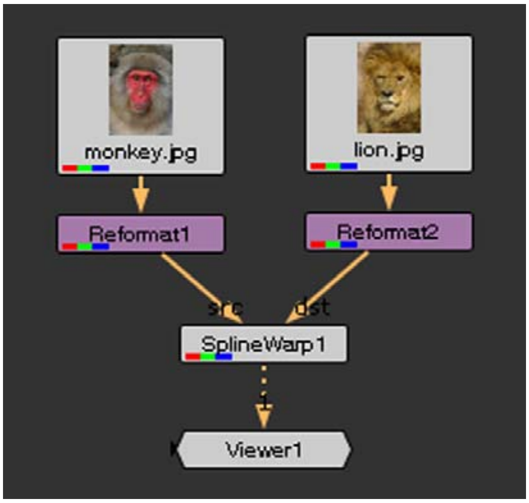
key. Then, scrub to the frame where you want the morph to end and set **distortion** to 1 (the destination image).

14. Animate the **blend** control in the same way. Scrub to the frame where the morph starts, and set **blend** to 0 (the source image). Click the animation button and select **Set key**. Scrub to the frame where the morph ends, and enter 1 (the destination image) as the **blend** value.

If you now play the sequence in the Viewer, you'll notice that the source image is morphed into the destination image.

To morph one image into another using the SplineWarp node:

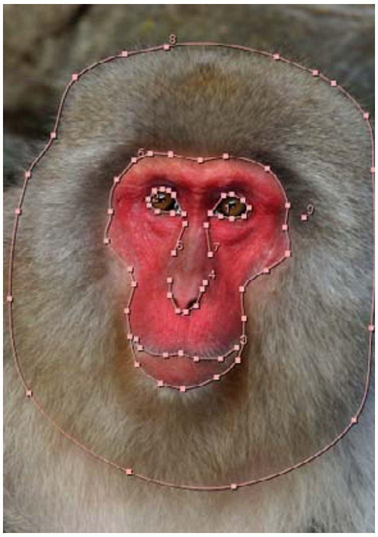
1. Select **Image** > **Read** to import the two images you want to morph together.
2. If the images do not have the same resolution, insert Reformat nodes after them. For more information, see "Reformatting Image Sequences" on page 139.
3. Select **Transform** > **SplineWarp** to insert a SplineWarp node into your script.
4. Connect the source image (the image you want to turn into another image) to the **src** input of the SplineWarp node, and the destination image (the image you want the source image turned into) to the **dst** input. Attach a Viewer to the SplineWarp node.



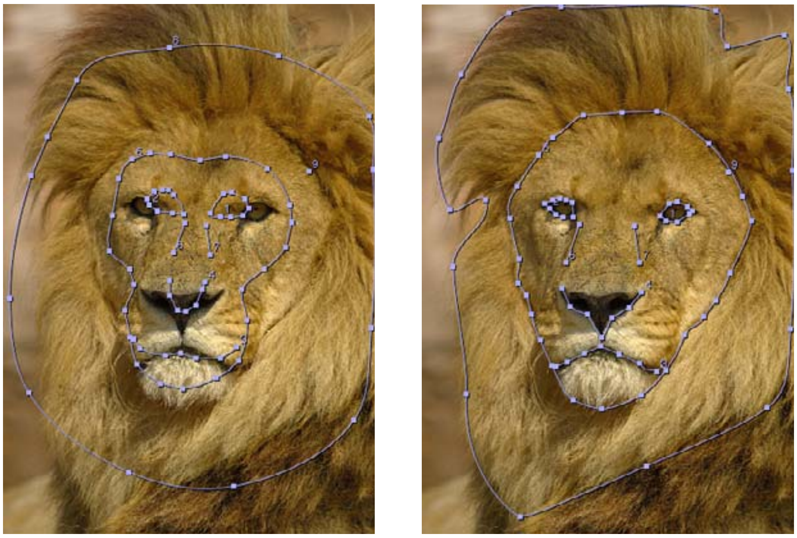
5. In the SplineWarp node's controls, set **show** to **src**. You should see the source image in the Viewer.
6. Identify some features that are similar in the source and the destination images. For example, if you are morphing together images of people or

animals, these features might include their eyes, noses and mouths as well as the outlines of their faces and heads.

7. In the Viewer, **Ctrl+Alt+click** (Mac users **Cmd+Alt+click**) to draw curves around the key features you identified in the previous step. For more information on how to create the curves, see "To warp an image using the SplineWarp node:" on page 393.



8. Click **copy** under **Source Curves** and **paste** under **Destination Curves**. This copies the source curves you created in the previous step into the destination curves, making them identical.
9. From the **show** pulldown menu, select **dst**. You should now see the destination curves and the image connected to the **dst** input.
10. Adjust the destination curves to conform to the features of the destination image.



11. In the SplineWarp node’s controls, select **blend** under **show**.
12. Scrub to the frame where you want the morph to begin. Bring the **distortion** slider down to 0 (the source image). Click on the animation button and select **Set key**. Then, scrub to the frame where you want the morph to end and set **distortion** to 1 (the destination image).
13. Animate the **blend** control in the same way. Scrub to the frame where the morph starts, and set **blend** to 0 (the source image). Click the animation button and select **Set key**. Scrub to the frame where the morph ends, and enter 1 (the destination image) as the **blend** value.

If you now play the sequence in the Viewer, you’ll notice that the source image is morphed into the destination image.

13 CREATING EFFECTS

Several nodes in Nuke let you create various effects on your input images. In this chapter, we describe three of these nodes: LightWrap, Glint, and Text. The LightWrap node lets you create background reflections on foreground elements. The Glint node can be used to create star filter effects on image highlights. The Text node is useful for adding text overlays on images, for example, when creating slates or scrolling credits.

Background Reflections on Foreground Elements

You can use the LightWrap node to create background reflections on foreground elements. The node creates a reflection of light around the edges of your foreground element by blending in whatever is in the background.

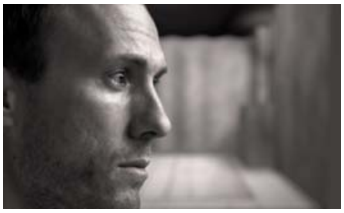


Figure 13.1: Using the LightWrap node to create background reflections on the edges of the foreground element: The composite without the LightWrap effect.

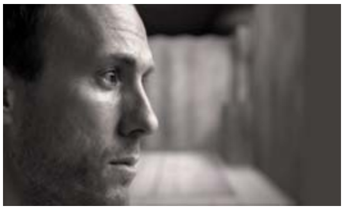


Figure 13.2: Using the LightWrap node to create background reflections on the edges of the foreground element: The composite with the LightWrap effect.

You may have noticed that objects filmed in front of very bright backgrounds have the appearance of softened edges as the light spills round from the background. When adding foreground layers onto a background plate, using the LightWrap node can dramatically improve the quality of your composite.

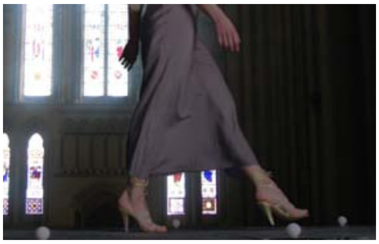


Figure 13.3: The composite without the LightWrap effect. Note the hard edges around the trailing hand and the bottom of the skirt.

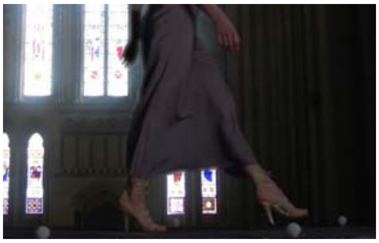


Figure 13.4: The composite with the LightWrap effect. Note how the foreground fits better into the background.

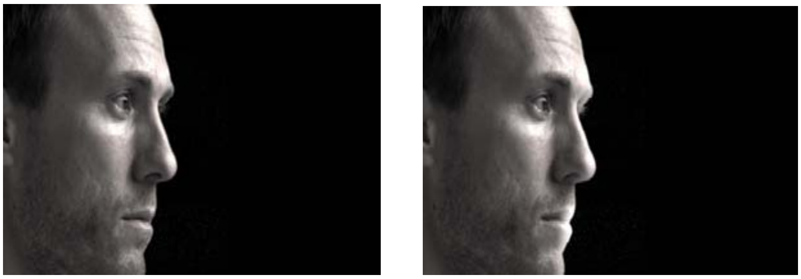
If you want to use LightWrap, you should apply it on your foreground element before you composite the foreground over the background with the Merge node.

To use the LightWrap node:

1. Select **Draw** > **LightWrap** to add a LightWrap node after your foreground and background images.
2. Connect your foreground element to input **A** of the LightWrap node, and the background image to input **B**.
3. Connect a Viewer to the output of the LightWrap node so you can see the effect of your changes.
4. Adjust the **Diffuse** and **Intensity** sliders to control both the spread and brightness of the reflections on the foreground element. These sliders need to be balanced out together. You may want to start by bringing Diffuse all the way down to better see what you are blending in from the background. Then, adjust Intensity before going back to the Diffuse slider and, if necessary, Intensity again until you are happy with the result.

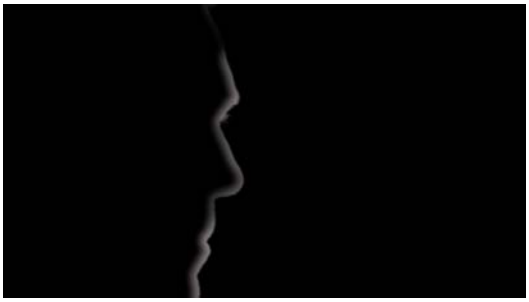
low diffuse

high diffuse

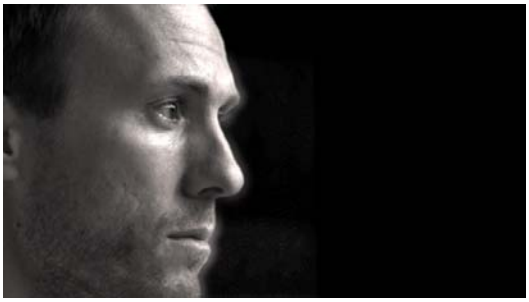


low intensity high intensity

5. If you want to create a uniform effect around the edges of the foreground rather than have the effect adjust itself according to the background, check **Disable luminance based wrap** on the **LightWrap** tab.
6. In case you don't want to merge the LightWrap effect with the foreground element in order to keep the LightWrap effect as a separate element, check **Generate wrap only** on the **LightWrap** tab.



7. By default, the LightWrap effect is only applied inside the foreground element's alpha. If you want to extend the effect outside it, making the element seem to glow, check **Enable Glow**.



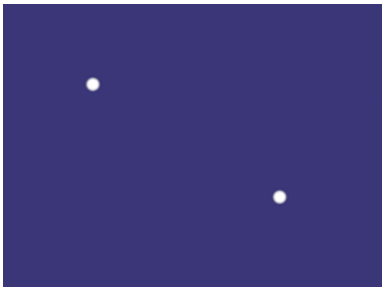
8. On the **Tweaks** tab, you can also adjust the following controls:
   - **FGBlur** to determine how much the foreground matte is blurred. The more blur, the more of the background is added to the foreground.

- **BGBlur** to control how much the background is blurred before it is merged with the foreground element.
- **Saturation** to adjust the saturation of the effect.
- **Luma Tolerance** to increase or decrease the luminance values of the effect.
- **Highlight Merge** to control how the foreground element is merged with the background. The default merge operation, called **plus**, adds the elements together, producing a glow effect.
- Check **Use constant highlight** to use a constant color of your choice rather than the background in the LightWrap effect. Select the color using the controls next to **Constant Highlights Color**.

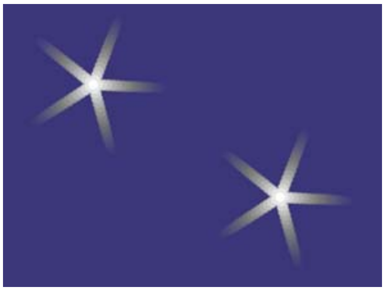
9. On the **CCorrect** tab, you can color correct the LightWrap effect produced.

Creating Star Filter Effects on Image Highlights

With the Glint node, you can create star-shaped rays around all the bright points in an image.



The original image with bright points.



The image after using the Glint node.

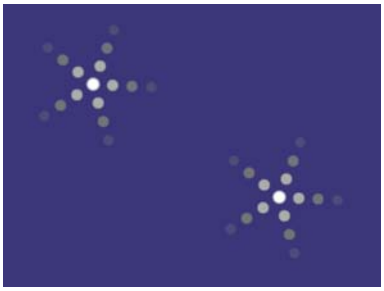
To use the Glint node:

1. Select **Draw** > **Glint** to add a Glint node after the image you want to add star-shaped rays to.
2. From the **channels** pulldown menu and checkboxes, select the channels to which you want to apply the effect.
3. In the **no. of rays** field, enter the number of rays you want coming out of the bright points in your image. For example, if you want to create five-pointed stars, enter 5.
4. To change the threshold for how bright the highlights in the input image need to be to cause the glint effect, adjust the **tolerance** slider. Only the pixels above the threshold will bloom with the effect.

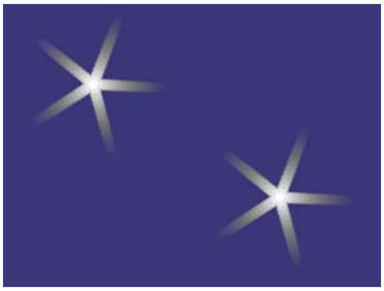
Low tolerance value.

High tolerance value.

5. To determine the length of the rays, adjust the **length** slider. To give every other ray a different length and determine that length, adjust the **odd ray length** slider.
6. To determine how many steps the rays are formed of, enter a value in the **steps** field. The more steps you use and the shorter the rays are, the smoother the rays become.

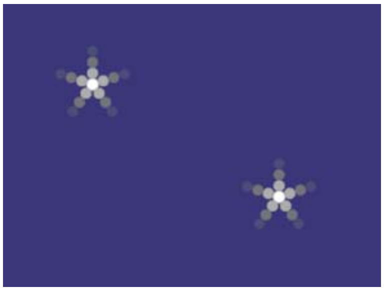


The steps value set to 3.

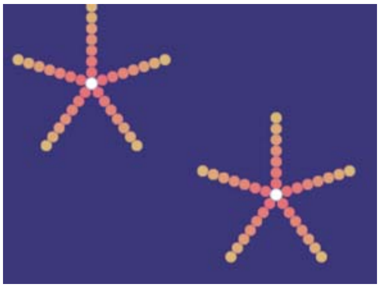


The steps value set to 5.

7. To rotate the star-shapes, adjust the **rotation** slider. Increasing the value rotates the rays clockwise, whereas decreasing the value rotates them counter-clockwise.
8. To change the color in the beginning of the rays near the center point of the stars, adjust the **from color** slider. To change the color in the end of the rays, adjust **to color**. By default the from color is set to white, and the to color to black.

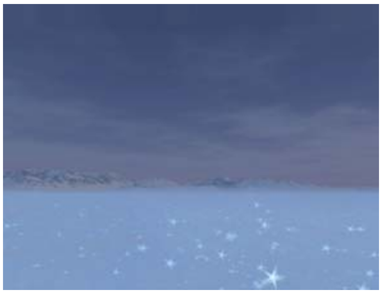


'From color' set to white, and 'to color' to black.

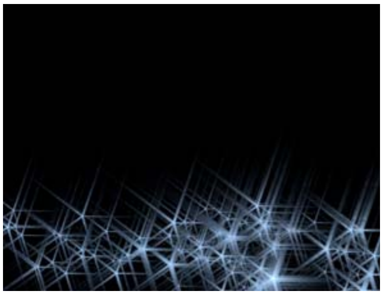


'From color' set to pink, and 'to color' to yellow.

9. If needed, you can also make the following adjustments:
   - If you want to change the aspect ratio of the stars, adjust the **aspect ratio** slider.
   - By default, the brightest image on the rays is used as the center point for the star. However, if you prefer the images forming the rays to be added up in forming the center point, uncheck **max**.
   - To only output the Glint effect without merging it into the original input image used to create it, check **effect only**.



Glint effect on an input image.



Glint effect without the input image.

   - To mask the shape that is used to create the rays, check **w** and select the mask channel from the pulldown menu.
   - To perform a gamma correction on the highlights that cause glint before the glint effect is applied, adjust the **gamma** slider.
   - To mask the glint effect, check **mask** and select a mask channel using the controls on the right.
   - To dissolve between the original input image and the full glint effect, adjust the **mix** slider.

Creating Text Overlays

Using Nuke's Text node, you can add text overlays on your images. You can simply type in the text you want to have displayed or use TCL expressions or TCL variables to create a text overlay. Text overlays can also be animated so that their properties, such as position, size, and color, change over time. These features make the Text node useful, for example, for creating slates or scrolling credits.

Creating a Text Overlay

To create a text overlay:

1. Select **Draw** > **Text** to create a Text node and connect a Viewer to it.
2. In the Text node properties, select the channels you want the text to appear in from the **output** controls.
3. If you want to multiply any channels by the drawn text so that they are set to black outside the text shape, select those channels using the **premult** controls.
4. From the **clip to** menu, select how you want to restrict the output image.
   - **no clip** - Do not restrict the output image.
   - **bbox** - Restrict the output image to the incoming bounding box.
   - **format** - Restrict the output image to the incoming format area.
   - **union bbox+format** - Restrict the output image to the combination of the incoming bounding box and format area.
   - **intersect bbox+format** - Restrict the output image to the intersection of the incoming bounding box and format area.
5. If you want to clear the affected channels to black before drawing on them, check **replace**. By default, **replace** is not checked and the text is drawn on top of the input image.
6. In the **message** field, enter the text you want to display, a TCL expression, a TCL variable, or a combination of these. For examples, see "Examples" on page 413.

   You should enter TCL expressions in square brackets, for example, [date].

   To begin a new line, press Return.

   To display special Unicode characters, such as foreign language characters and copyright signs, you can
   - use HTML named entities, such as &lt; to display <, or &copy; to display ©
   - use hex entities, such as &#x3c; to display <, or &#xa9; to display ©
   - use decimal entities, such as &#60; to display <, or &#169; to display ©

- type Unicode characters, such as < or ©, on your keyboard or cut and paste them from other applications. UTF-8 character encoding is used to store them in the control's value and in the saved Nuke script.

The above only work if the font you are using has the character you want to display in it.

Note *We recommend using the above entities rather than typing <, for example. This is because future versions of the Text node may interpret HTML mark-up. In HTML, some characters, such as the greater than and less than signs, are reserved. If you used these signs within your text now, future versions could mistake them for HTML mark-up.*

Note *If you're working in a stereoscopic project and want to split off a view on the **message** control, you can click the View menu and select **Split off [view]**, and specify a message per view. To view your results, you need to switch between your views using in the Viewer 's **view** selection, since the message control doesn't have separate entry fields for separate views.*

7. If you want to mask the effect of the Text operation, select the mask channel from the **mask** menu. To invert the mask, check **invert**. To blur the edges of the mask, check **fringe**.
8. To adjust the opacity of the text, use the **opacity** slider. The possible values run from 0 (invisible) to 1 (fully opaque).
9. To invert the inside and outside of the text shape, check **invert**.

Examples

The following table gives examples of TCL expressions, TCL variables, HMTL named entities, hex entities, and decimal entities you can use in the **message** field of the Text node.

Message	Prints
TCL expressions	
[date]	Week day, day, month, hh:mm:ss, and time zone. For example, **Thu Jan 15 14:22:20 GMT**.
[date %a]	Abbreviated week day name. For example, **Thu**.
[date %A]	Full week day name. For example, **Thursday**.
[date %b]	Abbreviated month name. For example, **Jan**.
[date %B]	Full month name. For example, **January**.
[date %d]	Day (01-31).
[date %D]	Date (dd/mm/yy). For example, **15/01/10**.

Message	**Prints**
[date %H]	Hour (00-23).
[date %I]	Hour (01-12).
[date %m]	Month (01-12).
[date %M]	Minutes (00-59).
[date %p]	AM or PM.
[date %r]	Time (12-hour clock). For example, **11:04:07 AM**.
[date %S]	Seconds (00-59).
[date %T]	Time (24-hour clock). For example, **14:06:54**.
[date %y]	Abbreviated year (00-99). For example, **10**.
[date %Y]	Full year. For example, **2010**.
[date %z]	Numeric time zone. For example, **-0800**.
[date %Z]	Time zone. For example, **GMT**.
[frame]	Frame number. For example, **23**.
[metadata]	List of all the keys in the incoming image metadata.
[metadata *key*]	Value of the key in the incoming image metadata. Replace *key* with the name of the key whose value you want to display. For example, you may be able to use **[metadata input/filename]** to display the name and location of the image file, or **[metadata input/ctime]** to display the timestamp for an input file.
[value root.name]	Script directory path and script name. For example, **Users/john/Nuke_scripts/myscript.nk**.
TCL variables	
$env(*ENVIRONMENT_VARIABLE*)	The value of the environment variable specified. Replace *ENVIRONMENT_VARIABLE* with an environment variable you have set. For example, you can use **$env(USER)** to display the user name (for example, **john**) on Mac OS X and Linux, or **$env(USERNAME)** to display it on Windows and Linux. For a list of environment variables specific to Nuke, see "Nuke Environment Variables" on page 614.
$version_long	The full version number of Nuke. For example, 6.2v1.
$threads	Number of render threads used to calculate images. This is in addition to a main thread used to update the graphical user interface (GUI).
HTML named entities	
&amp;	&
&apos;	'
&Aring;	Å

Message	Prints
&Aacute;	Á
&Acirc;	Â
&AElig;	Æ
&Agrave;	À
&Ccedil;	Ç
&copy;	©
&eacute;	é
&ecirc;	ê
&egrave;	è
&euml;	ë
&euro;	€
&gt;	>
&lt;	<
&Ntilde;	Ñ
&oslash;	ø
&otilde;	õ
&Ouml;	Ö
&ouml;	ö
&quot;	"
&reg;	®
&szlig;	ß
&Uuml;	Ü
&uuml;	ü
hex entities	
&#x23;	#
&#x25;	%
&#x26;	&
&#x2a;	*
&#x40;	@
&#x99;	™
&#x153;	œ
&#x161;	š

Message	Prints
&#x3c;	<
&#x3e;	>
&#xa9;	©
&#xe9;	é
decimal entities	
&#163;	£
&#169;	©
&#174;	®
&#191;	¿
&#234;	ê
&#223;	ß
&#224;	à

Tip *To get a list of all the TCL expressions you can use with **date**, type **x** on the Node Graph, set the script command dialog that opens to **TCL**, enter **date -h**, and click **OK**.*

Repositioning and Transforming Text

Using the on-screen widgets or the **Transform** controls on the Text node, you can reposition and resize the box that limits where the text is drawn and apply geometric transformations (including translations, rotations, scales, and skews) to the text.

To transform text:

1. The on-screen box limits the text inside a certain area of the frame. To reposition this area, drag the center of the box to a new location.

   To resize the area, drag the corners or edges of the on-screen box to a new location.

   Alternatively, you can expand the **Transform** controls in the Text node properties and enter new values for the **box** fields:

   - To define the left boundary of the box, adjust the **x** field.
   - To define the bottom boundary of the box, adjust the **y** field.
   - To define the right boundary of the box, adjust the **r** field.
   - To define the top boundary of the box, adjust the **t** field.

   Your text is wrapped inside the box you defined. However, parts of the text may appear outside the box if you have set **justify** to **baseline**, if a word is too long to fit in the box horizontally, or if there are too many lines of text to fit in vertically.

2. To translate, rotate, scale, or skew the text, use the transformation overlay. For more information on how to do this, see "Using the 2D Transformation Overlay" on page 226.

   Alternatively, you can adjust the **translate**, **rotate**, **scale**, and **skew** parameters in the Text controls.

   To reposition the center point of rotation and scaling, **Ctrl/Cmd**+drag the center of the transformation overlay to a new location, or adjust the **center** parameter in the Text node controls.

Fonts

By using the FreeType library, the Text node supports a large number of fonts, including

TrueType (.ttf) fonts and PostScript fonts (.pfa and .pfb).

To select a font:

1. In the Text controls, click on the folder icon next to **font**.

   The File Browser opens.
2. Browse to where you have stored the font and select it. To preview the font, click on the black arrow in the top right corner of the File Browser. Select **Open**.
3. Some font files contain more than one font face. If you are using such a font file, use the index field to the right of the **font** control to select the font face to use.

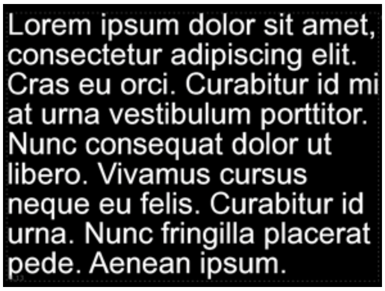
To adjust font size, spacing, and justification:

1. To adjust the size of the font, use the **size** slider. When **leading** is set to 0, this parameter also controls the spacing between each line of text.

   When rendering the font, the **size** parameter controls the font hinting used. Font hinting adjusts which pixels are interpolated to more clearly render a font. At small sizes and on low resolution output devices, it has a big impact on the legibility of the font. For best results, you should use the **size** parameter (rather than the **scale** parameter in the **Transform** parameter group) to control the size of the font and keep **scale** set to 1.
2. To increase or decrease the spacing between each letter, adjust the **kerning** slider. By using negative values, you can make the letters overlap.

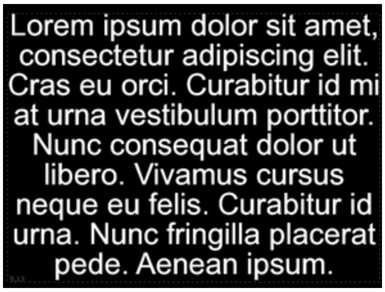
3. If your text overlay contains several lines of text, you can adjust the spacing between each line by using the **leading** slider. By using negative values, you can make the letters overlap.
4. From the left-side **justify** menu, select how you want to align the text horizontally:

- **left** - Align the text along the left edge of the on-screen text box. This leaves the right edge of the text ragged.



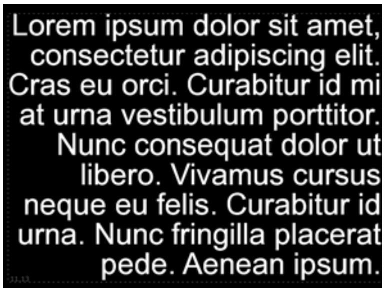
Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

- **center** - Align the text from the center of the on-screen text box. This leaves both edges of the text ragged.



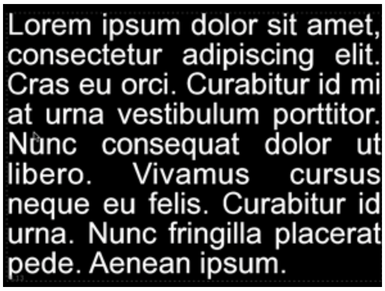
Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

- **right** - Align the text along the right edge of the on-screen text box. This leaves the left edge of the text ragged.



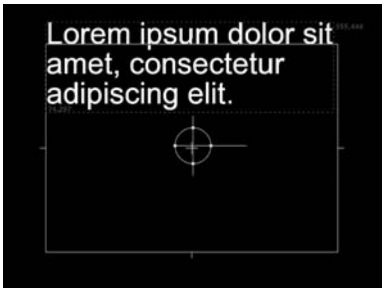
Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

- **justify** - Align the text both along the left and the right edge of the on-screen text box. This leaves no ragged edges. The justification is done by expanding the spaces between letters. If there are no spaces or the spaces get more than about three times wider than they were, letters are expanded.

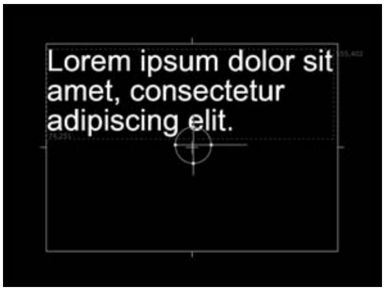

Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

5. From the right-side **justify** menu, select how you want to align the text vertically:

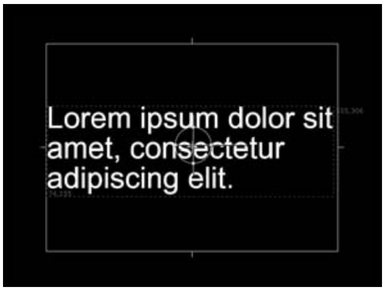
   - **baseline** - Align the baseline of the first line of text along the top edge of the on-screen text box. The baseline is the imaginary line upon which most letters rest. This option allows you to reliably line up the baseline of different fonts.


Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

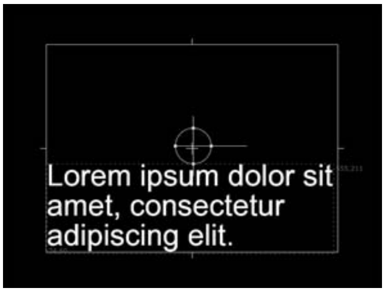
   - **top** - Align the text against the top edge of the on-screen text box.


Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

- **center** - Align the text from the center of the on-screen text box.


Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

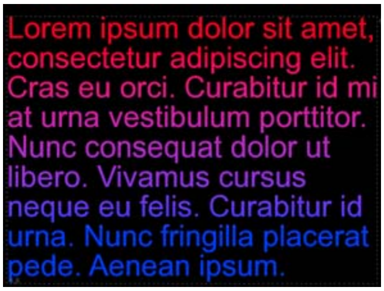
- **bottom** - Align the text against the bottom edge of the on-screen text box.


Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

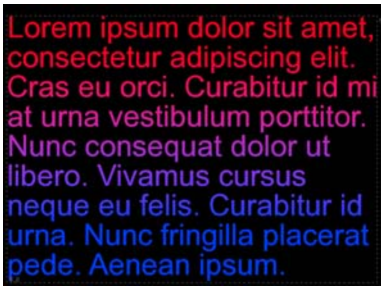
Changing the Text Color

To change the text color:

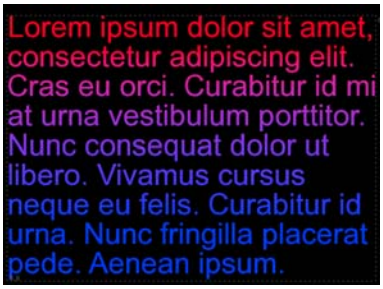
1. In the Text node controls, go to the **Color** tab.
2. Adjust the **color** parameter or click on the color picker button to select a color for the text.
3. If you want to create a color gradient across the text, select anything other than **none** from the **ramp**:
    - **linear** - the ramp changes linearly from one color into another.
    - **smooth0** - the ramp color gradually eases into the point 0 end. This means colors in the point 0 end are spread wider than colors in the point 1 end.
    - **smooth1** - the ramp color eases into the point 1 end. This means colors in the point 1 end are spread wider than colors in the point 0 end.
    - **smooth** - the ramp color gradually eases into both ends. This means colors in the point 0 and point 1 ends are spread wider than colors in the center of the ramp.


Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

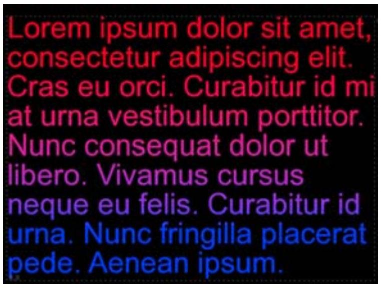
Linear ramp.


Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

Smooth ramp.


Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

Smooth0 ramp.


Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Cras eu orci. Curabitur id mi
at urna vestibulum porttitor.
Nunc consequat dolor ut
libero. Vivamus cursus
neque eu felis. Curabitur id
urna. Nunc fringilla placerat
pede. Aenean ipsum.

Smooth1 ramp.

4. Use the **color** parameter to choose a color for the ramp at the point 1 end (by default, the top end). Then, use **color 0** to choose the color for the ramp at the point 0 end (by default, the bottom end).
5. To adjust the spread and angle of the ramp, drag the on-screen **point 0** and **point 1** controls to a new location. You may need to press O twice on the Viewer to make these controls appear.

   Alternatively, you can enter new **x** and **y** values for the **point 1** and **point 0** parameters in the Text node properties.

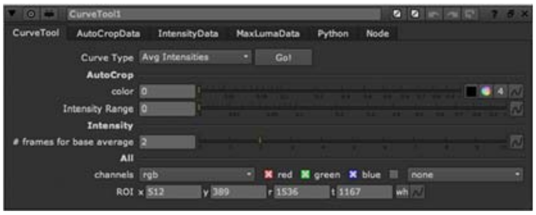
14 ANALYZING FRAME SEQUENCES

This chapter concentrates on the CurveTool node. The node analyzes an aspect of a frame sequence and creates an animation curve based on the analysis. You can then use the curve data to drive effects elsewhere. For instance, you can add matching flicker to a CG render.

Analysing and Matching Frame Sequences

You can use the CurveTool node to analyze four different aspects of your frame sequence, depending on which curve type you choose in the node controls:

- **AutoCrop** finds black regions (or any color you pick) around the edges of the frame sequence and tracks their size and position over time. This is useful for running a Crop node to remove unnecessary outer pixels and speed up the calculation.
- **Avg Intensities** is useful for obtaining the average pixel values in a frame sequence and then matching that intensity elsewhere. It takes the first value in the frame range and the next value selected, adds them together and divides by two, returning the average between the two. You might want to use it to match the background plate's fire flicker in the smoke in the foreground plate, for example.
- **Exposure Difference** analyzes the exposure changes in the frame sequence. It takes the first value in the frame range and the next value selected, and returns the difference between the two. You can use the results to match the same exposure elsewhere.
- **Max Luma Pixel** tracks the brightest and dimmest pixels in the frame sequence. This can be useful in the following case, for example. Let's say you have a night-time sequence depicting a person moving inside a dark house holding a flashlight, and want to add lens flare on the moving flashlight. Knowing where the brightest pixel is located over time allows you to match-move the lens flare and position it correctly without having to manually animate it.



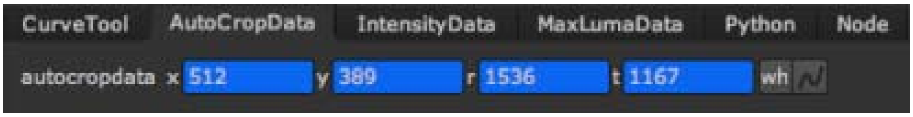
Curve Type: Avg Intensities
color: 0
Intensity Range: 0
# frames for base average: 2
channels: rgb ☒ red ☒ green ☒ blue ☐ none
ROI x 512 y 389 r 1536 t 1167

Figure 14.1: The CurveTool node properties panel.

Tip *If you are familiar with Shake, you may have used the PixelAnalyzer node. The CurveTool node is the Nuke equivalent of PixelAnalyzer.*

To crop black edges to eliminate unnecessary computation:

1. Choose **Image** > **CurveTool** to insert a CurveTool node after the image sequence you want to analyze.
2. Make sure a Viewer is connected to the CurveTool node.
3. In the CurveTool controls, select **AutoCrop** from the **Curve Type** menu.
4. Using the **color** parameters, select the color you want to track.
5. To control how far the color can deviate from the selected color and still be cropped off, use the **Intensity Range** slider.
6. From the **channels** menu and checkboxes, select the channels you want to analyze.
7. If you want to analyze an area in the frames rather than entire frames, define a region of interest either by dragging the edges of the frames to a new position in the Viewer, or by defining the area using parameters labeled **ROI**.
8. Click **Go!** to analyze the frames. This opens the *Frames to Execute* dialog.
9. In the dialog, define the frames to analyze. Enter the first frame, followed by a comma and the last frame. Click **OK**. Nuke starts analyzing the frame sequence.
10. You'll find the results of the analysis on the **AutoCropData** tab where the parameter values have turned blue to indicate they are animated over time. To see the animation curve, right-click on a parameter field and select **Curve editor**.



autocropdata x 512 y 389 r 1536 t 1167

Once Nuke has created the animation curve, you can copy the animation or any of its values into a Crop node, for example, to match the analyzed crop area there. **Ctrl/Cmd**+click on the animation button and drag and drop it to another parameter to create an expression linking the two.

To analyze the intensity of a frame sequence:

1. Select **Image** > **CurveTool** to add a CurveTool node in an appropriate place after the image sequence you want to analyze and match.
2. Connect a Viewer to the CurveTool.

3. In the node's controls, select **Avg Intensities** from the **Curve Type** pulldown menu.
4. Select the channels you want to analyze from the **channels** menu and checkboxes.
5. By default, the region of interest that is to be analyzed covers the entire frame. If you want to analyze a smaller area, resize and reposition the region of interest in the Viewer by dragging its edges to a new position. You can also resize the region of interest using the **ROI** parameters in the control panel.
6. In the **# frames for base average field**, enter the range of frames that each frame being analyzed is compared against. The frames are compared onwards from each frame analyzed. Thus, a value of 1 would compare each frame to the frame following it, whereas a value of 5 would compare each frame to the following 5 frames.

   The higher frame range you use, the more accurate and time-consuming the calculation becomes. However, a high frame range is not always needed. For analyzing and matching fire flicker, you'd probably want to go frame by frame, whereas removing flicker would require a wider frame range to ensure a good average is obtained as the result.
7. To analyze the sequence, click **Go!**. This opens the *Frames to execute* dialog.
8. In the dialog, specify the frame range you want to analyze and match. Enter the first frame, followed by a comma and the last frame. Click **OK**. Nuke now analyzes the frame sequence.
9. Move to the **IntensityData** tab where you'll find the results of the analysis. You'll notice that the parameter input fields have turned blue. This indicates that they are animated. To see the animation curve, right-click on the values and select **Curve editor**.

Once Nuke has created the animation curve, you can copy the animation or any of its values into a color correction node, for example, to match the analyzed intensity there. Ctrl/Cmd+click on the animation button and drag and drop it to another parameter to create an expression linking the two.

To analyze the exposure differences in a frame sequence:

1. Select **Image** > **CurveTool** to add a CurveTool node after the image sequence you want to analyze.
2. Add a Viewer after the CurveTool.
3. Under **Curve Type**, select **Exposure Difference**.
4. From the **channels** pulldown menu and checkboxes, choose the channels you want to analyze.

5. If you want to analyze an area in the frame rather than the entire frame, define a region of interest either by dragging the edges of the frame box to a new position in the Viewer, or by defining the area using parameters labeled **ROI**.
6. To analyze the sequence, click **Go!**. The *Frames to Execute* dialog opens.
7. Specify the frame range you want to analyze. Enter the first frame, followed by a comma and the last frame. Click **OK**. Nuke now performs the analysis.
8. You can find the results of the analysis on the **IntensityData** tab where the parameter input fields have turned blue to indicate they are animated. To see the animation curve, right-click on one of the input fields and select **Curve editor**.

Once Nuke has created the animation curve, you can copy the animation or any of its values into a color correction node, for example, to match the analyzed exposure there. **Ctrl/Cmd**+click on the animation button and drag and drop it to another parameter to create an expression linking the two.

To track the brightest and darkest pixels in a frame sequence:

1. Choose **Image** > **CurveTool** to add a CurveTool node after the image sequence you want to analyze.
2. Connect a Viewer to the CurveTool.
3. From the **Curve Type** menu, select **Max Luma Pixel**.
4. Click **Go!** to analyze the frame sequence. This opens the *Frames to Execute* dialog.
5. Define the frame range you want to analyze. Enter the first frame, followed by a comma and the last frame. Then, click **OK**. Nuke analyzes the frame sequence, tracking both the position and the values of the brightest and darkest pixels.
6. You can find the results of the analysis on the **MaxLumaData** tab. You'll notice that the input fields have turned blue to indicate that they are animated over time. To see the animation curve, right-click on an input field and select **Curve editor**.

Once Nuke has created the animation curve, you can copy the animation or any of its values into another node to match that node's effect with the brightest or darkest pixels in the frame sequence. **Ctrl/Cmd**+click on the animation button and drag and drop it to another parameter to create an expression linking the two.

15 3D COMPOSITING

Nuke's 3D workspace allows you to setup a 3D composite for camera moves, set replacement, and other applications where you need to simulate a "real" dimensional environment.

Although the 3D workspace has many potential uses, you're most likely to use it—at least initially—to create pan-and-tile scenes. These are scenes with 2D image planes arranged into a curved shape, and then rendered out through an animated camera to give the illusion of a seamless environment.

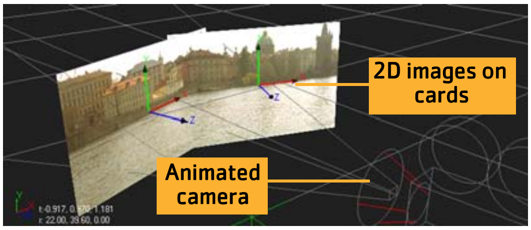

2D images on cards
Animated camera

Figure 15.1: Simple pan-and-tile scene.

Overview

This chapter explains how to set up a 3D scene in Nuke, and how to add objects and cameras in the 3D workspace. You'll also see how to texture objects, transform objects and cameras, and render out scenes for use in other areas of your script.

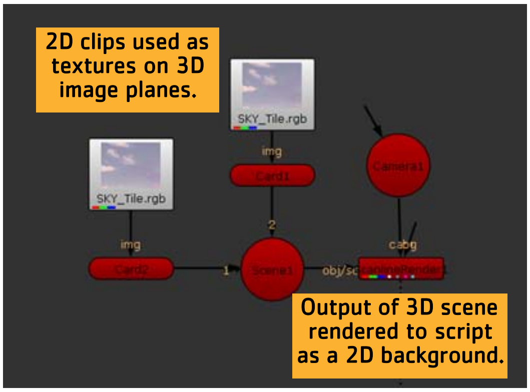

2D clips used as textures on 3D image planes.
SKY_Tile.rgb
SKY_Tile.rgb
img
img
Card1
Camera1
2
Card2
1
Scene1
cam
bg
obj/sc
Output of 3D scene rendered to script as a 2D background.

Figure 15.2: Script with 2D and 3D operators.

The 3D objects in Nuke appear as round shapes to differentiate them from objects that perform 2D operations. As shown above, you can mix 2D and 3D objects together in the node tree. For example, you can texture a 3D object with a 2D clip, or take the rendered output from a 3D scene and use it as a 2D background.

Setting Up a Scene

Each 3D scene includes the following objects: a Scene node, a Camera node, one or more geometry nodes (i.e., card, sphere, obj), and a ScanlineRender node. Examples of 3D scenes are shown in Figure 15.2 and Figure 15.3. In the example shown in Figure 15.2, the Scene node receives the output from two geometry nodes (*Card1* and *Card2*) and sends the composite of those objects to the ScanlineRender node, where the output is converted back to 2D.

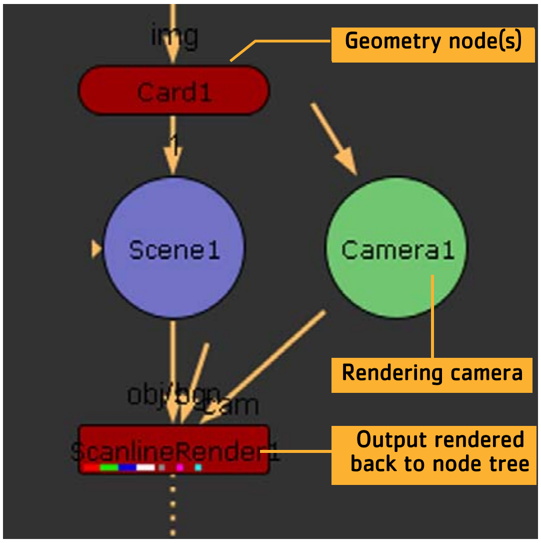


Figure 15.3: Core nodes for a 3D composite.

Your script may contain multiple Scene nodes, cameras, and 3D render nodes. All 3D objects loaded in the Properties Bin will appear in the 3D Viewer, regardless of whether they are connected to the same Scene node.

The Scene Node

Regardless of its location in your script, the Scene node is the highest-level node in the scene hierarchy because it references all the elements in a 3D workspace—all the geometric objects, cameras, and materials.

To add a Scene node:

Choose **3D** > **Scene** from the toolbar.

The ScanlineRender Node

Every Scene node in a script should be connected to a ScanlineRender node, which tells Nuke to render the results of the scene. The ScanlineRender node also allows you to toggle between a 2D and 3D view of the scene.

To add a ScanlineRender node:

1. Select the Scene node.
2. Choose **3D** > **ScanlineRender** from the toolbar.
3. Connect the **obj/scn** input to a Scene or geometry node.
4. Connect the **cam** input to the main camera.
5. Connect the optional **bg** input to composite a background image into the scene.
6. Press **Ctrl+I** (Cmd+I on a Mac) to open a new Viewer to display the output of the ScanlineRender node.

When an image is connected to the bg input, its resolution becomes the output resolution for the ScanlineRender node.

The Camera Node

Cameras may be connected to either the Scene node or the ScanlineRender node. The camera connected to the ScanlineRender node is the camera used for rendering.

1. Choose **3D** > **Camera** to insert a camera node.
2. Drag an output connector from the Camera node to a Scene node or connect the Camera node to a ScanlineRender node's cam input.

When connecting cameras for the 3D scene, the camera you want to use for rendering should be connected to the ScanlineRender node, like this:

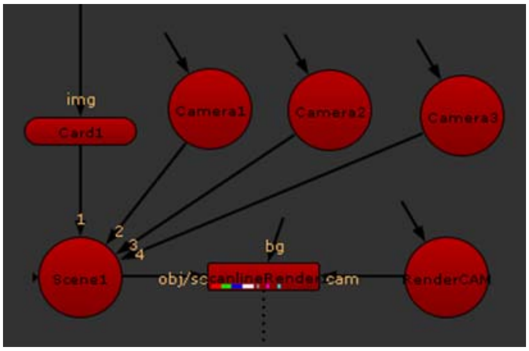
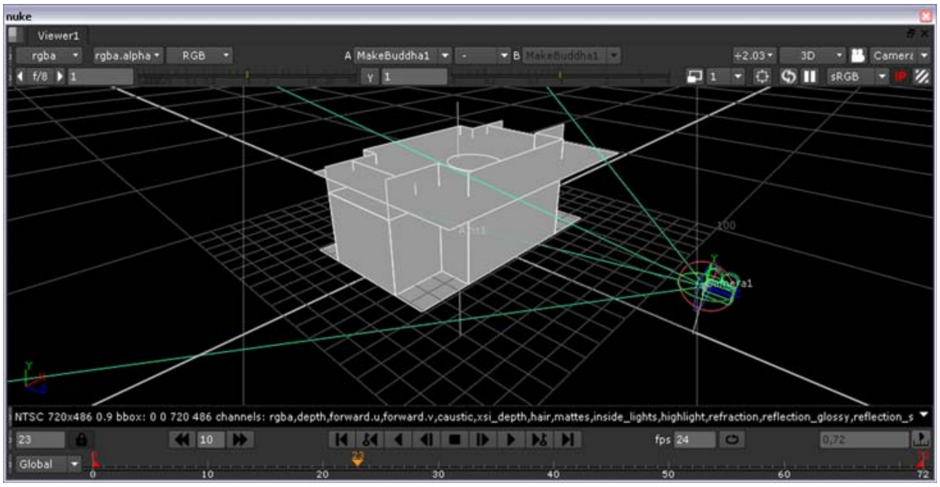


Figure 15.4: Connecting cameras to the scene.

Any additional cameras should be connected to the Scene node. When you have multiple cameras associated with a 3D scene, you can switch between them by choosing the viewing camera from the list at the bottom of the Viewer. See the next section, "Using the 3D Viewer", for more information.

Using the 3D Viewer

When you have a 3D setup in your script, any Viewer window can toggle between the 2D and 3D display modes. The 2D mode shows the result of a rendered scene, the 3D mode shows the perspective from one camera in the scene.



```
NTSC 720x486 0.9 bbox: 0 0 720 486 channels: rgba,depth,forward.u,forward.v,caustic,xsi_depth,hair,mattes,inside_lights,highlight,refraction,reflection_glossy,reflection_s
```

Figure 15.5: The 3D Viewer.

When you do not have a Camera node in your script, the 3D Viewer uses default views (see Figure 15.6 for the list of options). These views are similar to different cameras that you can look though, but they don't appear as objects that you can manipulate in the scene.

Tip *To put you into the scale, the Nuke 3D world is measured in centimeters (that is, not inches or pixels).*

Switching to the 3D Viewer:

Open a Viewer and press **Tab** or **V** to toggle between the 2D and 3D modes— or select the view you want from the list at the top right corner of the Viewer window.

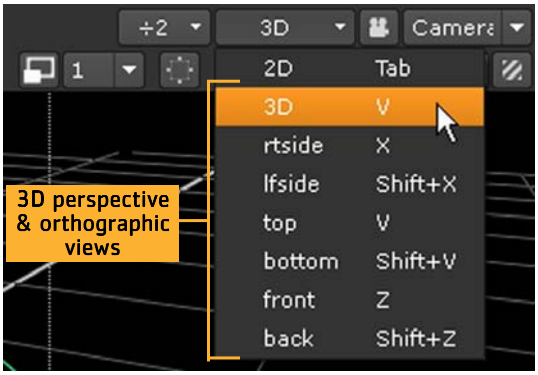

÷2
3D
Camera
1
2D Tab
3D V
rtside X
lfside Shift+X
top V
bottom Shift+V
front Z
back Shift+Z
3D perspective & orthographic views

Figure 15.6: Switching to the 3D Viewer.

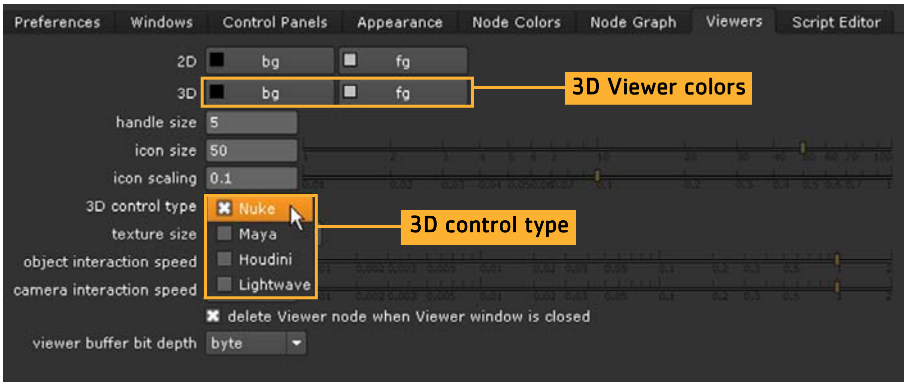
The "built-in" views give you different perspectives on your 3D scene. You can quickly switch between the views by pressing the hotkeys for right view (**X**), left view (**Shift+X**), top view (**C**), bottom (**Shift+C**), front (**Z**), back (**Shift+Z**), and three-quarter perspective (**V**).

To navigate in the 3D Viewer:

- *Dolly*: Press **Alt** and **middle-mouse-button** drag.
- *Pan*: Press **Alt** and **left-mouse-button** drag.
- *Tilt*: Press **Ctrl/Cmd** and **left-mouse-button** drag.
- *Spin*: Press **Ctrl/Cmd** and **left-mouse-button** drag.
- *Roll*: Press **Ctrl/Cmd+Shift** and **left-mouse-button** drag.
- *Look through camera*: Select a camera object, press **H**.
- *Fit the scene*: Press **F** to fit the entire 3D scene within the Viewer.

To change the 3D Viewer display properties:

1. Open the Preferences window (**Shift+S**), and select the **Viewers** tab.


Preferences
Windows
Control Panels
Appearance
Node Colors
Node Graph
Viewers
Script Editor

2D bg fg
3D bg fg
handle size 5
icon size 50
icon scaling 0.1
3D control type ☒ Nuke ☐ Maya ☐ Houdini ☐ Lightwave
texture size
object interaction speed
camera interaction speed
☒ delete Viewer node when Viewer window is closed
viewer buffer bit depth byte

3D Viewer colors

3D control type

Figure 15.7: 3D Viewer properties.

2. Make the desired changes to the 3D **bg** and **fg** colors.
3. From the **3D control type** list, select the navigation control scheme you want to use (Nuke, Maya, Houdini, or Lightwave).
4. Click **Save Prefs**.

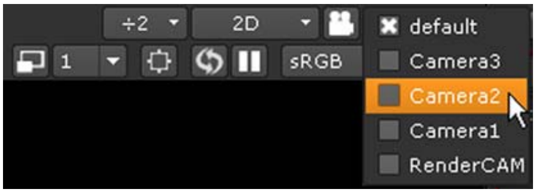
Note *The 3D control type also affects the mouse button assignments for panning and zooming in the node graph and 2D Viewers.*

To look through a camera:

1. Press **V** to make sure you are looking through the 3D perspective view, and not one of the orthographic views.
2. Select the camera in the Viewer or select the camera's node in the workspace. Then press **H** (home).

   or

   From the 3D Viewer window, select the camera from the list in the top right corner.


÷2
2D
1
sRGB
☒ default
☐ Camera3
☐ Camera2
☐ Camera1
☐ RenderCAM

Note *This selection does not change the camera used for rendering. This changes only the camera to "look through" for the current 3D Viewer.*

Cameras in the current data stream automatically appear in the list of cameras you can select. To select a camera that doesn't appear in list, double-click the camera node to open its panel, and it will be added to the list.

To lock the 3D camera view

You can choose to lock the 3D view to the selected camera or light. You can toggle between the unlocked and locked modes by clicking the **3D view lock** button. You can also activate the interactive mode by **Ctrl**/**Cmd**+clicking the **3D view lock** button:

- **unlocked**: to freely move in the 3D view without restrictions
- **locked:** to lock your movement to the camera or light you've selected in the drop-down menu on the right side of the **3D view lock** button.
- **interactive**: the camera or light values will change according to your movement in the Viewer. When the interactive mode is on, you can use the plus (+) and the minus (-) keys to change the translate values of the camera or light you've selected. In other modes, these keys will zoom your view in and out.

3D Scene Geometry

Nuke includes several options for inserting 3D geometry into your scenes. You can create primitive shapes, such as cards, cubes, and spheres, as well as import models created in other 3D applications.

These are the types of objects you can include in a Nuke 3D scene, and each objects is represented by a 3D node in the script:

- Cards
- Cubes
- Cylinders
- Spheres
- OBJ (Wavefront) objects
- Axes
- Cameras
- Lights

Working with Cards

A *card* is the simplest type of object you can add to a scene (and probably the type you will use most often). It's merely a plane onto which you can map a texture—typically a clip you are using as part of a pan-and tile setup.

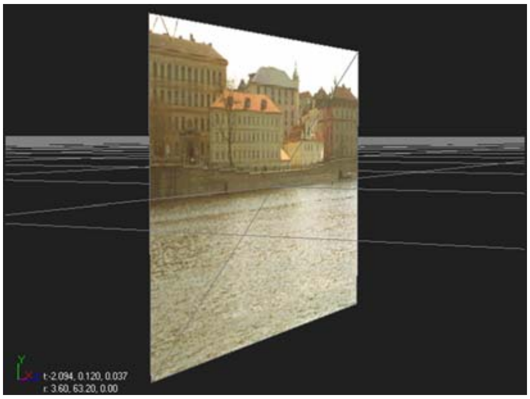


Figure 15.8: A card object.

A card object may be deformed as a bilinear or bicubic object with controls contained in the card's parameters. You can also insert other 3D nodes, such as ProceduralNoise or RadialDistort, to change the card geometry.

Card nodes have extended bicubics (bicubics with more control points). They allow you to subdivide a card, giving you finer control for warping an area. You can subdivide the card into an evenly spaced grid or pick a location to add a row, column, or both.

To add a card object:

1. Click **3D** > **Geometry** > **Card** to insert a Card node.
2. Drag the Card node's **img** pipe to the Read node that has the image you want to apply to the card.
3. Connect the Card node to the appropriate Scene node to add it to the 3D scene.
4. Use the card object's transform controls to manipulate the position, scale, and rotation of the card in 3D space. For more information, see "Transforming from the Node Properties Panel" on page 459.

Deforming Card objects

The Deform tab on the Card panel lets you convert the card into a mesh surface that may be pulled and reshaped.

A *bicubic deformation* offers the greatest degree of surface elasticity. You can add any number of control points on the card and translate these points and their tangents in any direction. The control point tangents exert a magnetic-like influence over the objects surface.

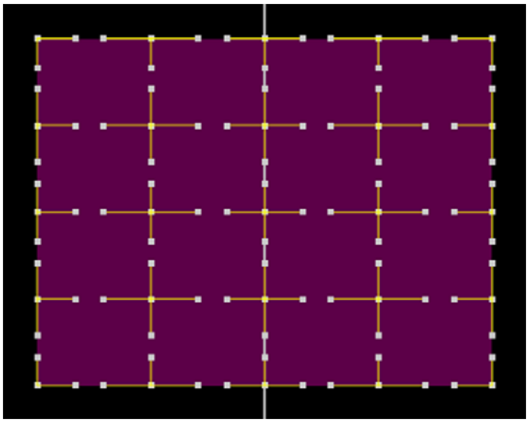
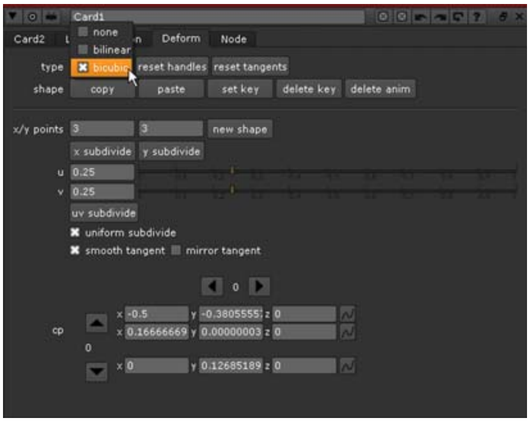


Figure 15.9: The Card node can have any number of control points you can translate.

To deform a Card object:

1. Double-click the Card node to open its controls.
2. Go to the **Deform** tab, and select the mesh type for the deformation: **bilinear** or **bicubic**.



Card1
Card2
Deform
Node
☐ none
☐ bilinear
reset handles
reset tangents
copy
paste
set key
delete key
delete anim
new shape
x subdivide
y subdivide
uv subdivide

type ☒ bicubic
x/y points 3 3
u 0.25
v 0.25
☒ uniform subdivide
☒ smooth tangent ☐ mirror tangent
0
cp 0
x -0.5 y -0.3805555: z 0
x 0.16666669 y 0.00000003 z 0
x 0 y 0.12685189 z 0

3. By default, the card has three control points on the x axis, and three on the y axis. To add more control points, do any of the following:
   - Enter new values in the **x/y points** fields and click the **new shape** button. For example, to create a shape with 4 points on the x axis and 6 on the y axis, change the **x points** value to 4 and the **y points** value to 6, and click **new shape**.
   - To evenly subdivide the current shape in the x or y directions, click the **x subdivide** or **y subdivide** buttons. This adds one control point between every existing control point in the chosen direction. The **x/y**

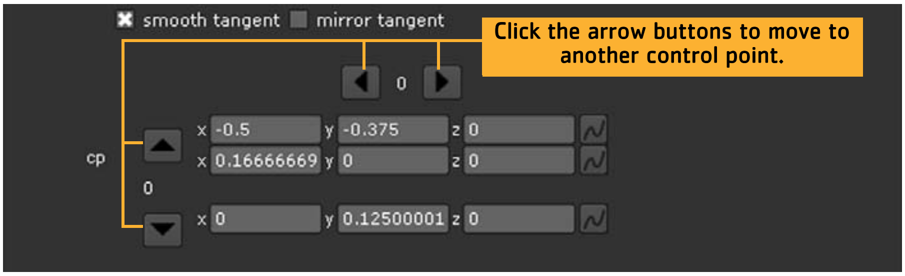
points fields are also updated to reflect the current number of control points.

- To add one row or column of control points, adjust the **u** or **v** slider. The **u** slider specifies the position of new columns, and the **v** slider the position of rows. In the Viewer, a yellow cross marker indicates the position of the new row or column. You can also move the cross marker by dragging it to a new position in the Viewer. The **u** and **v** sliders' values are updated as you move the marker. When you are happy with the position, click the **uv subdivide** button. A row or column is added in the position you specified. Clicking the button again has no effect, because there is already a subdivision at the specified position.

4. If you selected **bicubic** under **type**, you can adjust the way control point tangents behave when you are making your changes to the card. Do any of the following:
    - To have the original tangents adjusted to create a more uniform subdivision when you are using **x subdivide**, **y subdivide**, or **uv subdivide**, check **uniform subdivision**. If you do not check this, Nuke maintains the original tangents.
    - You can move the tangents in the Viewer by clicking and dragging. If you want to move a tangent together with the opposite tangent so that the two tangents form a continuous line, check **smooth tangent**. To break the tangent from the opposite tangent and move the tangent alone, uncheck **smooth tangent**.
    - To change the length of the opposite tangent to always match the length of the tangent you are moving, check **mirror tangent**. If you do not check this, the opposite tangent length is not changed.
5. Drag the points displayed in the mesh to deform the card.

To translate the control points and tangents:

1. If necessary, double-click on the Card node to display its controls, and go to the **Deform** tab.
2. Only the controls for the selected point are displayed in the bottom of the Card properties panel. To translate another point, you can select a new point in the Viewer or use the arrow buttons in the bottom of the Card controls to move to the next control point.



☒ smooth tangent ☐ mirror tangent

cp 0

x -0.5 y -0.375 z 0
x 0.16666669 y 0 z 0
x 0 y 0.12500001 z 0

Click the arrow buttons to move to another control point.

3. To translate the control points and their tangents:
   - Increment or decrement the numbered **x**, **y**, and **z** fields. For each control point, the controls for translating the point itself are shown on top of the controls for translating the tangents.
   - Or drag on any control point or tangent to translate it relative to the current angle of view.

Working with Cubes

A *cube* is the familiar six-sided polyhedron. You can transform any of its sides (and, of course, texture it with clips).

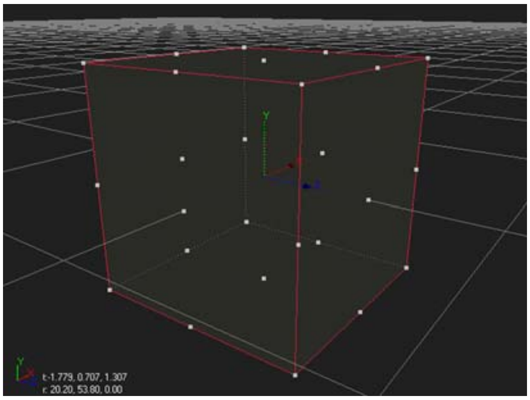


Figure 15.10: A cube object.

To add a cube:

1. Click **3D** > **Geometry** > **Cube** to insert a Cube node.
2. Drag the Cube node's **img** pipe to the Read node containing the clip you want to use as a texture.
3. Drag one of the Scene node's numbered pipes to the Cube node to place the cube in the scene.

4. Use the cube object's transform controls to manipulate the position, scale, and rotation of the cube in 3D space. For more information, see "Transforming from the Node Properties Panel" on page 459.
5. Translate any of the cube's sides to alter its shape.

To translate a cube's sides:

1. If necessary, double click on the Cube node to display its parameters (and thereby select the object in the scene).
2. Increment or decrement the **cube** fields. (Assuming a positive z view of the object, **x** refers to the left side; **y**, the bottom side; **n**, the back side; **r**, the right side; **t**, the top side; and **f**, the front side.)

   Or drag on any side order to translate it relative to the current angle of view.

Working with Spheres

A *sphere* is the familiar globe-shaped polyhedron. You can control its geometric resolution, or face count (and, of course, texture it with clips).

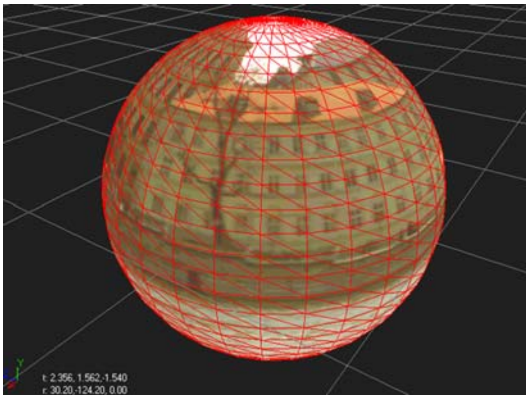


Figure 15.11: A sphere object.

To add a sphere:

1. Click **3D** > **Geometry** > **Sphere** to insert a Sphere node.
2. Drag the Sphere node's **img** pipe to Read node containing the clip you want to use as a texture.
3. Drag one of the Scene node's numbered pipes to the Sphere node to place the bicubic object in the scene.

4. Use the sphere object's transform controls to manipulate the position, scale, and rotation of the sphere in 3D space. For more information, see "Transforming from the Node Properties Panel" on page 459.

Adjusting geometric resolution

By default, a sphere has 30 rows and 30 columns. You can, however, increase or decrease either number as appropriate. For example, the figure below shows a sphere whose geometric resolution has been decreased to 2 rows and 4 columns (making it, in effect, an octahedron).

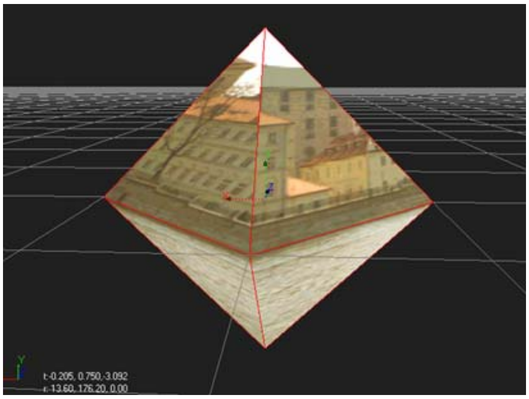


Figure 15.12: An octahedron generated with a low-resolution sphere.

To adjust a sphere's geometric resolution:

1. If necessary, double click on the Sphere node to display its parameters (and thereby select the object in the scene).
2. Increment or decrement **rows** field to adjust the number of latitudinal divisions on the sphere.
3. Increment or decrement **columns** field to adjust the number of longitudinal divisions on the sphere.

Working with OBJ Objects

You can import into a Nuke scene 3D objects from other software programs which have been saved out in the OBJ (Wavefront) format. You cannot manipulate OBJ objects at the vertex level from inside Nuke, but you can texture and transform them.

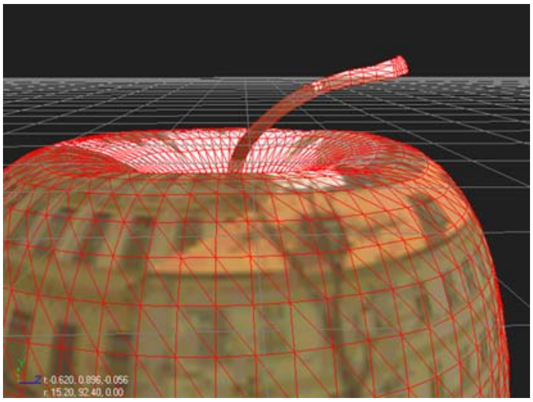


Figure 15.13: An imported OBJ object.

To import an OBJ object:

1. Click **3D** > **Geometry** > **ReadGeo** to insert a ReadGeo node.
2. In the ReadGeo parameters, click the **file** field's folder icon. The file navigation dialog appears.
3. Navigate to the OBJ file, then click **Open**. Nuke reads in the OBJ file.
4. Drag the ReadGeo node's **img** pipe to the Read node containing the clip you want to use as a texture.
5. Drag one of the Scene node's numbered pipes to the ReadGeo node to place the OBJ object in the scene.

3D Selection Tools

Using the 3D selection tools in the top right corner of the Viewer, you can select nodes, vertices and faces on a 3D object. Additionally, you can select individual 3D objects in the 3D view, which is handy when you have more than one object in a node.

Selecting nodes

You can select nodes with the Node selection tool. This is the default selection tool, and corresponds to the familiar way of selecting 3D object nodes in the Viewer.

Selecting vertices on a 3D object

You can select vertices on a 3D object using the Vertex selection tool. To save selections into your script, you need to use the GeoSelect node. You can find the 3D selection tools in the top right corner of the Viewer.

Note *The selection mode is set by the Viewer—changing geometry nodes translates your selections to the new object.*

To select vertices in the Viewer

1. Attach a 3D object to the Viewer and activate the Viewer's 3D mode.
2. Click the **Vertex selection** tool to activate it. It is disabled by default.
3. Select vertices on the object by dragging a marquee over the vertices you want to select.
4. By default, a new selection replaces an existing one, but you can also add and remove from existing selection:
   - To add to existing selection, hold down **Shift** while selecting.
   - To remove from existing selection, hold down **Shift+Alt** while selecting.
5. You can also include hidden items by selecting the **occlusion test** button. Unselect it if you don't want to include occluded vertices in your selection. Select if you want to select both visible and occluded vertices.

To save or restore selected vertices using the GeoSelect node

With the default Viewer 3D selection tools you can only make a temporary selection. To save or restore a selection, you can use the same 3D selection tools in the Viewer with the GeoSelect node:

1. Create a GeoSelect node (**3D** > **Modify** > **GeoSelect**) and attach a 3D object to its input. The GeoSelect control panel needs to be open in order for you to use the node for selecting vertices.
2. Click the **Vertex selection** tool from the 3D selection panel.
3. Select vertices on the object by dragging a marquee over the vertices you want to select.
4. By default, a new selection replaces an existing one, but you can also add and remove from existing selection:
   - To add to existing selection, hold down **Shift** while selecting.
   - To remove from existing selection, hold down **Shift+Alt** while selecting.
5. In the GeoSelect control panel, you can also uncheck the **selectable** box to freeze your selection so it can't be changed. Checking the box again enables selection again and you can continue selecting vertices as normal.
6. With the **display** and **render** controls you can select the display option you want for your object. For more information on these controls, see "Object Display Properties" on page 456.

7. When you're happy with your selection, click **save selection** to store the vertices in the GeoSelect node.
8. The vertices are saved by the GeoSelect node and can be restored by clicking **restore selection**.

Matching Position, Orientation and Size to 3D Selection

You can match the position, orientation and size of a 3D object, such as a Card node, to your selected vertices in another object. Do the following:

1. Click the **snap** menu on the control panel of the 3D object you want to move and select **Match selection position**. The object is now positioned to match the selected vertices.
2. If you select **snap** > **Match selection position, orientation**, your 3D object will be positioned and aligned according to the 3D vertex selection.
3. If you select **snap** > **Match selection position, orientation, size**, your 3D object will be positioned, aligned and scaled according to the selected vertices.

Parenting to Axis Objects

An axis object works as a null object by adding a new transformational axis to which other objects may be parented. Even when objects already have their own internal axes, it's sometimes useful to parent in a separate axis.

For example, the Axis node has been parented to the other objects in the scenes (the two image planes and the camera). The result is an axis object which globally controls the scene. Rotating it, for example, rotates all objects in the scene, as the figure below shows.

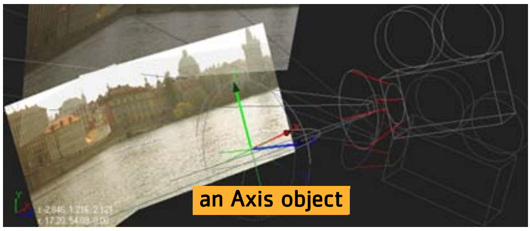
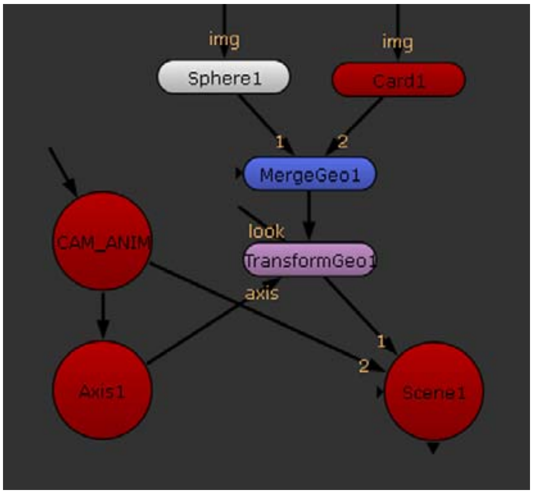


Figure 15.14: Rotating an entire scene with an axis object.

Tip *To move several objects together, you can also merge them using a MergeGeo node (see "Merging Objects" on page 443) and then control them using a TransformGeo node (see "Using the TransformGeo Node" on page 460).*

To add an axis object:

1. Click **3D** > **Axis** to insert an Axis node.
2. To create the parent relationships, connect the Axis node all object nodes you wish to control with the Axis transformation controls.


img
img
Sphere1
Card1
1
2
MergeGeo1
look
TransformGeo1
axis
CAM_ANIM
Axis1
1
2
Scene1

To create a nested transformational hierarchy, chain additional Axis nodes to the first one you inserted. For example, you could create a hierarchy of three Axis nodes to control rotation, scale, and transform.

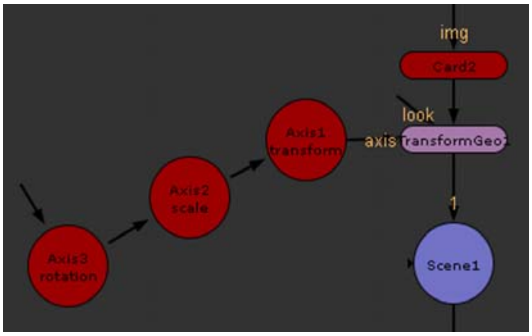

img
Card2
look
axis
TransformGeo1
Axis1
transform
Axis2
scale
Axis3
rotation
1
Scene1

Figure 15.15: Creating a nested transformational hierarchy.

In the above example, *Axis3* (rotation) is connected to *Axis2* (scale). *Axis2* is connected to *Axis1* (transform), and *Axis1* is connected to the TransformGeo node(s) that you want to affect. With the Axis nodes connected in this manner, their transformation data ripples down the chain and is added to the settings of the TransformGeo node.

Merging Objects

With the MergeGeo node, you can merge your 3D objects together to process all of them at the same time. For example, after merging your objects, you can use a Transform node to move the objects together, or add an ApplyMaterial node to apply a global material to them (note that this will override any individual materials applied to the geometry before it was merged).

To merge your 3D objects:

1. Select **3D** > **Modify** > **MergeGeo** to insert a MergeGeo after the 3D objects in your script.
2. Connect the objects you want to merge to the MergeGeo node's inputs.

   You can now process all the objects you connected to the MergeGeo node together.

Object Material Properties

The nodes under the Shader menu let you define the material attributes of geometric objects in a scene, including the quality of light reflected back to the camera from an object's surface. Using these nodes, you can control what material your objects seem to be made of.

You can also add several Shader nodes one after the other to produce more complex effects. For this, you should use the unlabeled inputs on the Shader nodes.

The material property settings you apply affect the render output of the scene.

You can insert 3D shader nodes in the following places in your scripts:

- between the 2D image you're using for the surface texture and the 3D object node that creates the surface.
- after the 3D object nodes using the ApplyMaterial node. Connect the geometry (for example, a Sphere node) to the ApplyMaterial node, and the materials (for example, a BasicMaterial node) to the ApplyMaterial node's mat input. This is a good way to apply a global material to all objects.

You can use the **map** connectors to input a mask image to limit the effect of the material change.

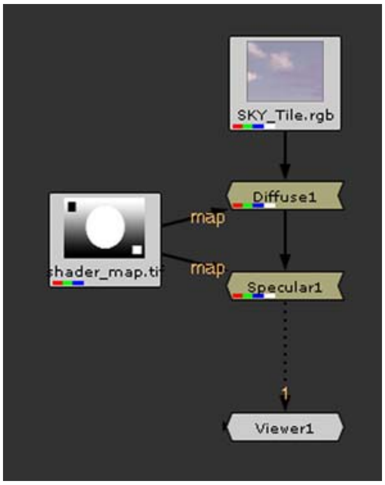


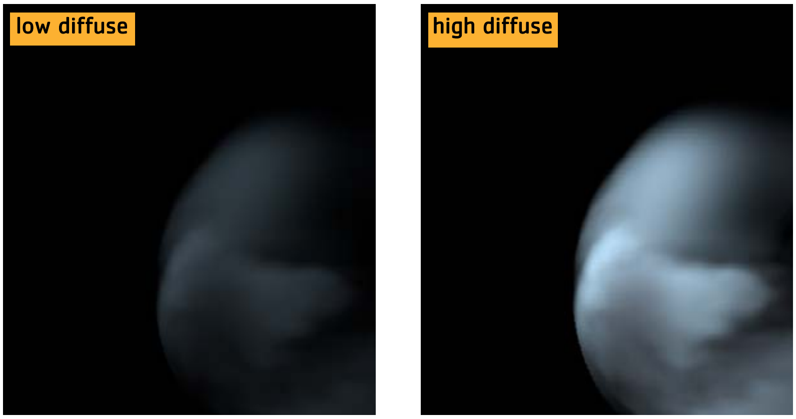
Figure 15.16: Diffuse and Specular nodes.

Note *To see the effect of changes to an object's material properties, transparency and lighting must be enabled for OpenGL rendering. Press **S** over the 3D Viewer, and check **transparency** and **headlamp** on the **3D** tab.*

To define material properties:

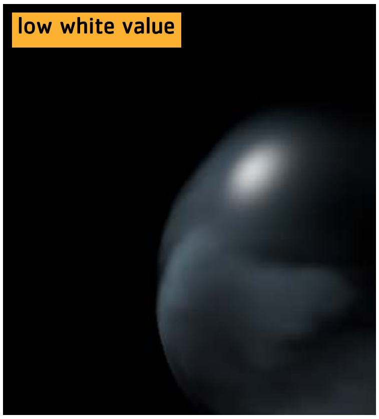
- Choose **3D** > **Shader** > **Diffuse** to insert a Diffuse node. This node lets you adjust the color of the material when illuminated. The material appears darker as the surface points away from the light, as the light is not falling on it.

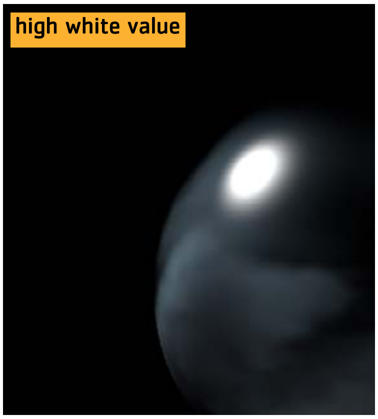
  Adjust the **white** slider in the Diffuse panel to control the diffuse color. By default, this is in grayscale, but you can adjust the individual r, g, and b values. The higher the value, the brighter the material.



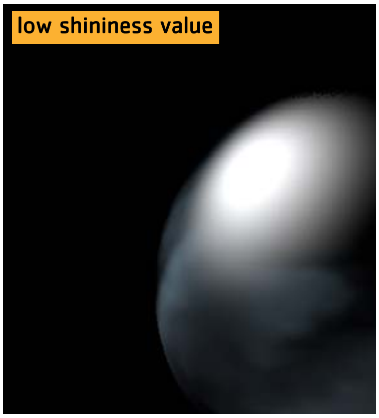
- Choose **3D** > **Shader** > **Specular** to insert a Specular node. You can use this node to control how bright and wide the highlights on the material seem. The location of the viewpoint is significant: the specular highlights are the brightest along the direct angle of reflection.

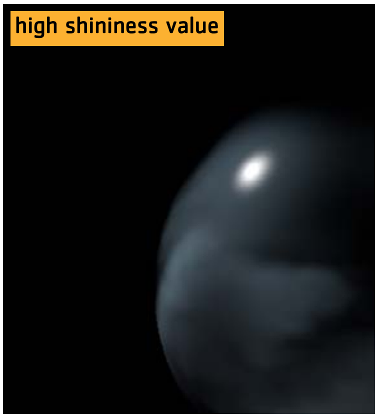
Adjust the **white** slider to control the brightness of the specular highlight. The higher the value, the shinier the material seems.





To control the width of the highlights, adjust the **min shininess** and **max shininess** sliders.





Adjust **shininess channel** to control how the input channels are used to map the black and white values to the minShininess and maxShininess parameters when a **mapSh** input is connected. Choose **red** to use the red channel for the mapping, **green** to use the green channel, **blue** to use the blue channel, **luminance** to use the luminance, or **average rgb** to use the average of the red, green, and blue channels.

- Choose **3D** > **Shader** > **Emission** to insert an Emission node. You can use this node to simulate lamps or other sources that emit light. Adjust the **emission** slider to change the brightness of non-illuminated areas for the surface. The higher the value, the more light the material seems to emit and the brighter it appears.

- Choose **3D** > **Shader** > **Phong** to insert a Phong node. This node uses the Phong algorithm to provide more accurate shading and highlights.

  The Phong node has several map inputs you can use to mask the effect of the node. You can use:
  - **mapD** to modulate the diffuse component,
  - **mapS** to modulate the specular component,
  - **mapE** to modulate the emission component, and
  - **mapSh** to modulate the shininess value.

  You can adjust the following sliders in the node's controls:
  - **color** to change the material color.
  - **emission** to change the color of the light the material emits.
  - **diffuse** to control the color of the material when illuminated.
  - **specular** to control how bright the highlights on the material seem.
  - **shininess** to control how shiny the material appears.
  - **min shininess** and **max shininess** to set the minimum and maximum shininess values. If you haven't connected an image to the **mapSh** input of the node, the average of these values is used as the shininess value for the material.
  - **shininess channel** to control how the input channels are used to map the black and white values to the minShininess and maxShininess parameters when a **mapSh** input is connected. Choose **red** to use the red channel for the mapping, **green** to use the green channel, **blue** to use the blue channel, **luminance** to use the luminance, or **average rgb** to use the average of the red, green, and blue channels.
- Choose **3D** > **Shader** > **Basic Material** to insert a BasicMaterial node. This node is a combination of the Diffuse, Specular, and Emission nodes, allowing you to control all three aspects of the material with a single node.

  Like the Phong node, the BasicMaterial node has several map inputs you can use to mask the effect of the node. You can use:
  - **mapD** to modulate the diffuse component,
  - **mapS** to modulate the specular component,
  - **mapE** to modulate the emission component, and
  - **mapSh** to modulate the shininess value.

  In the node's controls, you can adjust the following parameters:
  - **emission** to change the color of the light the material emits. Note that when you have an image connected to the unlabeled input of the BasicMaterial node and adjust this value, you need to look at the rendered 2D image to see the effect of your changes. Changing the emission value does not have any effect in the 3D Viewer.

- **diffuse** to control the color of the material when illuminated.
- **specular** to control how bright the highlights on the material seem.
- **min shininess** and **max shininess** to set the minimum and maximum shininess values. If you haven't connected an image to the **mapSh** input of the node, the average of these values is used as the shininess value for the material.
- **shininess channel** to control how the input channels are used to map the black and white values to the minShininess and maxShininess parameters when a **mapSh** input is connected. Choose **red** to use the red channel for the mapping, **green** to use the green channel, **blue** to use the blue channel, **luminance** to use the luminance, or **average rgb** to use the average of the red, green, and blue channels.

Projecting Textures onto Objects

You can use the UVProject and the Project3D nodes to project texture images onto your 3D objects. This way, you can add detail, surface texture, or color to your geometry, making the geometry more realistic and interesting.

The UVProject node changes the uv values of the vertices whereas the Project3D node is a material shader.

Projecting Textures with the UVProject Node

The UVProject node sets the uv coordinates for the object, allowing you to project a texture image onto the object. If the object already has uv coordinates, this node replaces them.

To use the UVProject node:

1. Select **3D** > **Modify** > **UVProject** to insert a UVProject node anywhere after the 3D object you want to modify.
2. Attach a Viewer to the node to see your changes.
3. In the node's controls, use the **display** drop-down menu to select how you want to view your object in the Viewer while making changes to it.
4. Connect an Axis or a Camera node to the UVProject node's axis/cam input. If you connect an Axis node, project the texture uv coordinates onto the object using the axis transform values (that is, translation, rotation, scale, etc.). If you connect a Camera node, do a similar projection as with the axis but also use the camera lens information, such as the aperture.
5. Adjust the following parameters:

- From the **projection** drop-down menu, select the projection type. Usually, it's best to select a type that's close to the object's surface shape. For example, if your object is a sphere, like a football or a planet, select **spherical**.
- From the **plane** drop-down menu, select the projection direction: **XY**, **YZ**, or **ZX** to project the texture image along the z, x, or y axis. This pulldown menu is only available if you selected **planar** as the projection type.
- From the **project on** drop-down menu, select **both**, **front** or **back** depending on whether you want to project the texture on the front face of the object, its back face or both. The front face of an object is the one facing the camera and similarly the back face is the one furthest away from the camera.
- Check **view frustum culling** if you want the UVProject node to affect only the vertices inside the camera view frustum. Any vertices outside the view frustum will not be affected and they still keep their original uv coordinates. Uncheck if you want the node to affect all vertices.
- To mirror the texture uv coordinates in the horizontal direction, check **invert u**. To mirror them in the vertical direction, check **invert v**.
- To scale (stretch or squash) the texture uv coordinates in the horizontal direction, adjust the **u scale** slider. To scale them in the vertical direction, adjust the **v scale** slider. The higher the value, the more the texture is stretched.
- To change the name of the attribute that's used as the vertex's uv coordinates to find the image pixel, enter a name in the **attrib name** field.

Projecting Textures with the Project3D Node

The Project3D node projects an input image through a camera onto the 3D object.

To use the Project3D node:

1. Select **3D** > **Shader** > **Project3D** to insert a Project3D node after the image you want to project. Connect a Camera node to the Project3D node's cam input.
2. Insert a 3D geometry node (for example, a sphere) after the Project3D node.
3. Attach a Viewer to the 3D geometry node to see your changes.
4. In the node's controls, use the **display** pulldown menu to select how you want to view your object in the Viewer while making changes to it.

5. From the **project on** pulldown menu, select to project the image on either the front facing, back facing, or both polygons.
6. To extend the input image at its edges with black, check **crop**. To extend the image with the edge colors, uncheck **crop**.

Replacing Material Channels with a Constant Color

The FillMat node lets you replace selected material channels with a constant color. Typically, you would use this node to make one object hold out the others. When you set the

FillMat color to 0, it acts as a "3D cookie cutter" and makes a black hole where the material would otherwise be.

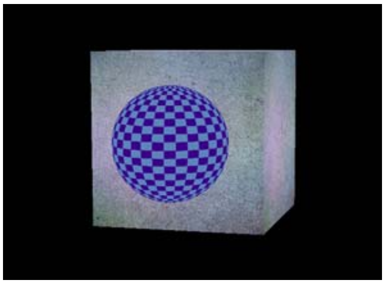


Figure 15.17: A sphere in front of a cube.

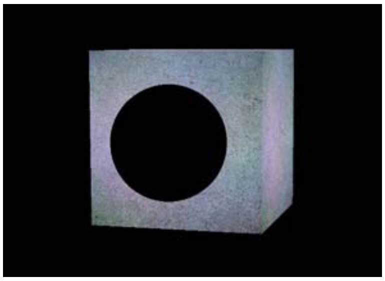


Figure 15.18: The same scene with the sphere material's rgba channels set to black using the FillMat node.

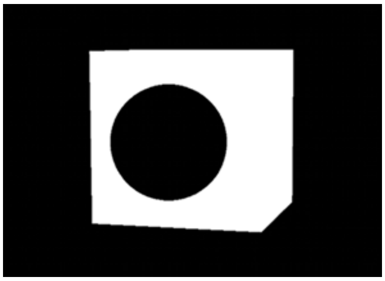
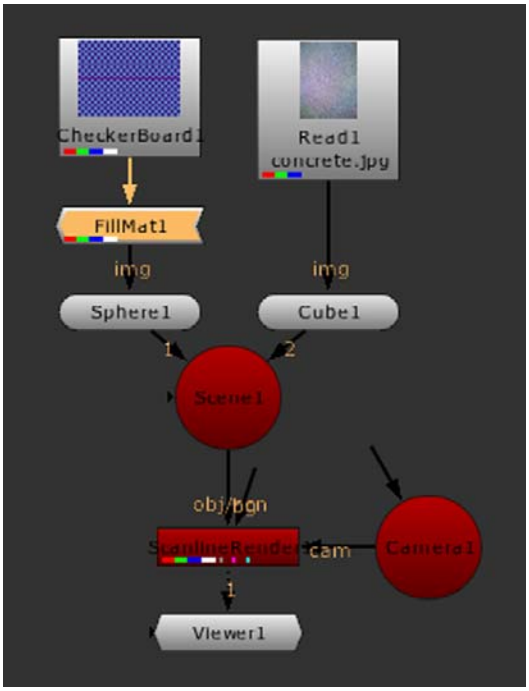


Figure 15.19: The alpha channel after applying the FillMat node.

This is similar to using a black Constant node as the input texture. However, the advantage of using the FillMat node is that you can easily apply it to the alpha channel in addition to the rgb channels. Another advantage is that the FillMat node doesn't break the shading sequence, so you can insert it after other material nodes in your node tree.

To replace selected material channels with a constant color:

1. Select **3D** > **Shader** > **FillMat** to insert a FillMat node between the 2D image you're using for the surface texture and the 3D object node that creates the surface.


CheckerBoard1
Read1
concrete.jpg
FillMat1
img
img
Sphere1
Cube1
1
2
Scene1
obj/bgn
ScanlineRender1
cam
Camera1
1
Viewer1

2. In the FillMat controls, use the **channels** controls to select the channels you want to replace with a constant color.
3. Use the **color** control to select the constant color. By default, this is set to black (0).

Merging Shaders

With the Shader menu's MergeMat node, you can combine two shader nodes together, using compositing algorithms like **none**, **replace**, **over**, and **stencil**. The MergeMat node is particularly useful for combining multiple Project3D nodes, allowing you to composite 2D images projected onto the 3D geometry atop each other.

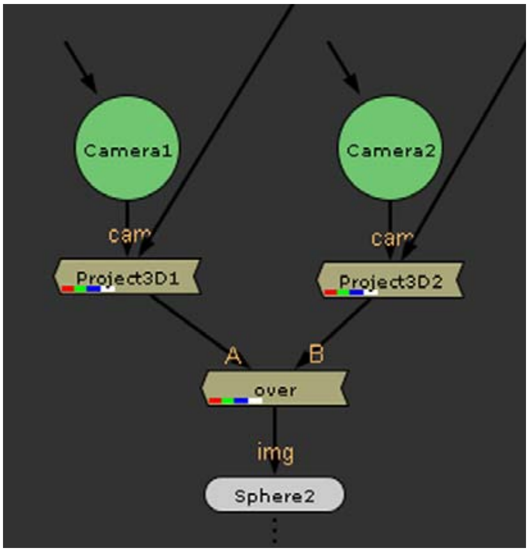
Merging Two Shader Nodes

You connect the shader nodes to the MergeMat node's A and B inputs. A refers to the foreground element, and B to the background element.

To merge two shaders:

1. Select **3D** > **Shader** > **MergeMat** to add a MergeMat (over) node after the two shader nodes you want to combine.
2. Connect the MergeMat node to the **img** input of the 3D object you want to project the images on.

   For example, if you wanted to combine two Project3D nodes and composite their results onto a sphere, your node tree would look something like the following:



3. For operations (such as **over**) that need an alpha channel (mask), select which channel to use for the alpha from the **Alayer** pulldown menu.
4. From the **operation** pulldown menu, select how you want to composite the results of the two shader nodes together:
   - to only use input B in the composite, select **none**.
   - to only use input A in the composite, select **replace**.
   - to composite input A over input B using a mask, select **over**.
   - to use input B outside the mask area, select **stencil**.
   - to use input B inside the mask area, select **mask**.
   - to add input B to input A, select **plus**.
   - to use input A if it is greater than input B or else use input B, select **max**.
   - to use input A if it is less than input B or else use input B, select **min**.

Merging a Material with the Objects Behind

The Shader menu's BlendMat node sets how the pixels colored by the material it is applied to combine with the pixels from objects behind. It is like the MergeMat node, but instead of blending with another material, it blends with whatever is rendered behind in the 3D scene.

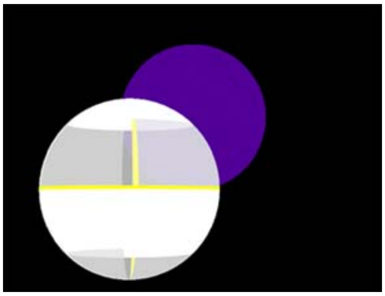


Figure 15.20: Without the BlendMat node.

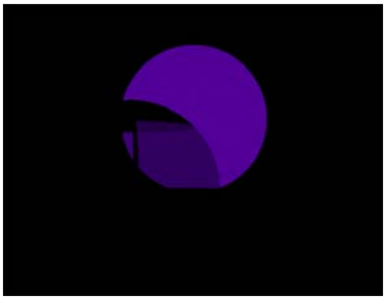
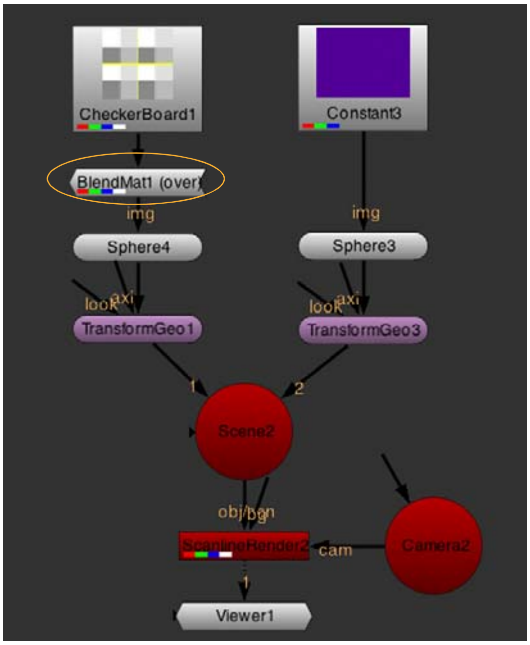


Figure 15.21: With the BlendMat node applied to the checkered sphere (that has a checkered alpha channel) and the BlendMat **operation** set to **stencil**.

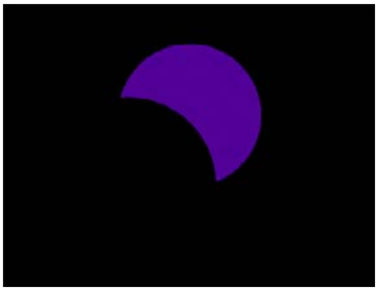
To merge a material with the objects behind:

1. Select **3D** > **Shader** > **BlendMat** to add a BlendMat node after the material you want to merge with the background pixels.
2. Connect the BlendMat node to the **img** input of the 3D object you want to project the material on.

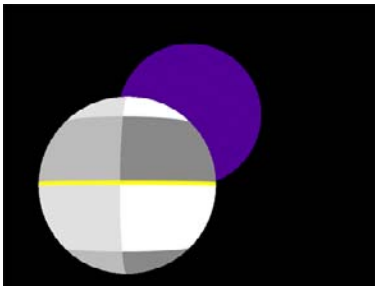

CheckerBoard1
Constant3
BlendMat1 (over)
img
img
Sphere4
Sphere3
look
axi
look
axi
TransformGeo1
TransformGeo3
1
2
Scene2
obj/scn
bg
ScanlineRender2
cam
Camera2
1
Viewer1

3. From the **channels** pulldown menu, choose the channels you want to affect.
4. From the **operation** pulldown menu, select how you want to composite the BlendMat node's input material and the background pixels together:

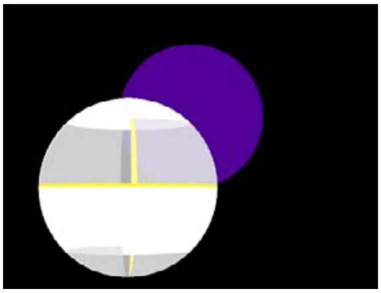
- to set the material to black, select **none**.



- to show the material where the material and the background overlap, select **replace**.



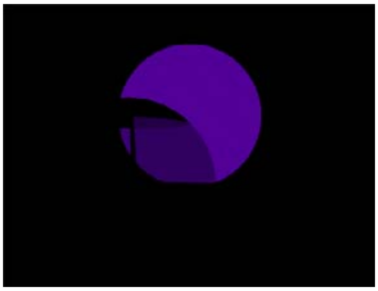
- to composite the material over the background pixels according to the material's alpha, select **over**.



- to show the background pixels where the material's alpha is black, select **stencil**. Where the material's alpha is white, the material is set to black.

  For this to work, the BlendMat node needs to process the alpha channel, so set **channels** to **rgba**.

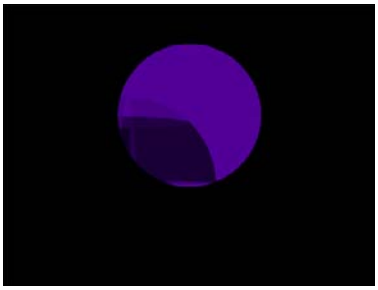
  This operation is the opposite of **mask**.



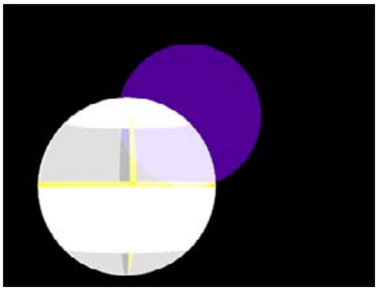
- to show the background pixels where the material's alpha is white, select **mask**. Where the material's alpha is black, the material is also set to black.

  For this to work, the BlendMat node needs to process the alpha channel, so set **channels** to **rgba**.

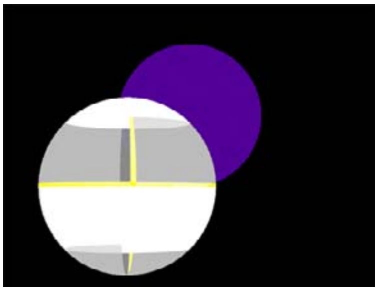
  This operation is the opposite of **stencil**.



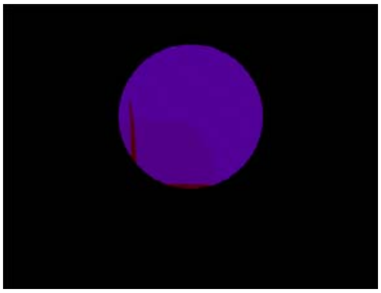
- to add the background pixels to the material, select **plus**.



- to use the material if its pixel values are greater than the background pixels or else use the background pixels, select **max**.



- to use the material if its pixel values are less than the background pixels or else use the background pixels, select **min**.

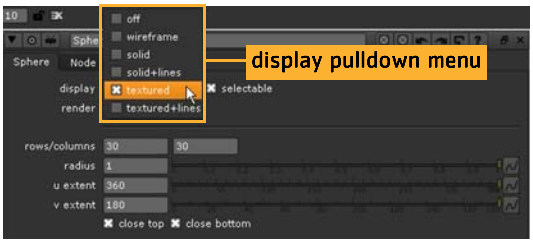


Object Display Properties

You can adjust the display characteristics of all geometric objects in a scene. These settings don't affect the render output of the scene; these are for display purposes only in the 3D Viewer.

To edit an object's display attributes:

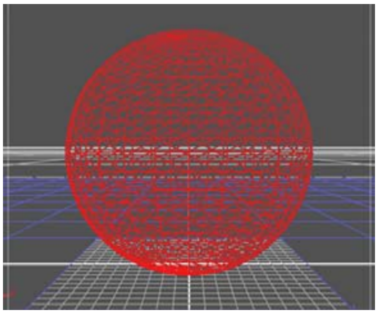
1. Double click on the object's node to display its parameters.
2. From the **display** list, choose the display type that you want for the object.


10
Sphe
Sphere
Node
off
wireframe
solid
solid+lines
textured
textured+lines
render
display pulldown menu

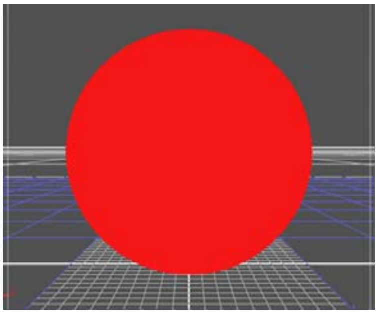
display ☒ textured
☒ selectable
rows/columns 30 30
radius 1
u extent 360
v extent 180
☒ close top ☒ close bottom

These are how each of the display options appear:

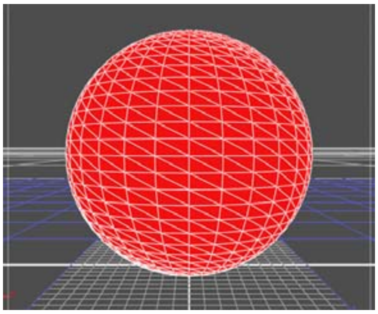
- **wireframe** displays only the outlines of the object's geometry.



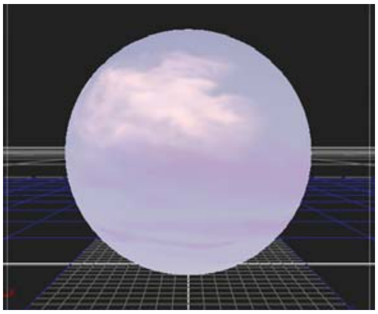
- **solid** displays all geometry with a solid color.



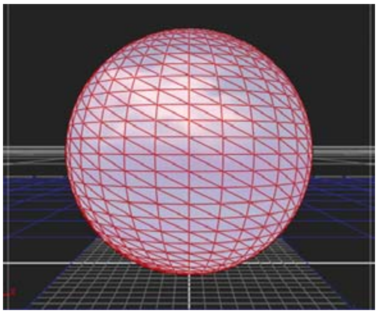
- **solid + lines** displays the geometry as solid color with the object's geometry outlines.



- **textured** displays the only the surface texture.



- **textured + lines** displays the wireframe plus the surface texture.



Transforming Objects

Transform operations include moving, scaling, rotating the objects in your 3D scene. When an object node is active, you can enter specific transform settings in the node parameters, or directly manipulate the object with the transform handles displayed in the 3D Viewer. You can also link transform parameters to imported track or camera data, or control the transforms with animation curves.

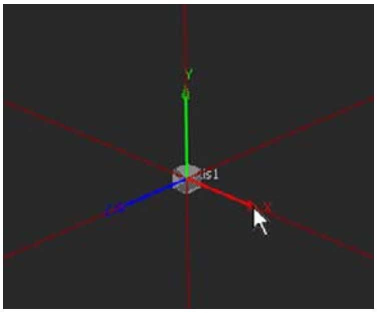
Cameras, geometry objects, and lights have their own set of transform controls built-in.

Using the Transform Handles

Transform handles appear when a 3D object with transform capabilities is loaded into the Properties Bin. The colors of the handles correspond to the axes available in 3D space: red transforms the x-axis, green transforms the y-axis, and blue transforms the z-axis.

To move an object with the transform handles:

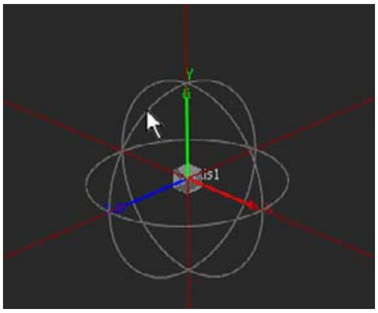
- Drag an object to move it on any axis.



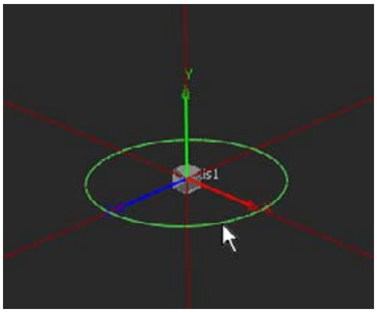
- **Shift**+drag to constrain movement to one axis.

To rotate an object with the transform handles:

- **Ctrl**+drag (Mac users **Cmd**+drag) to rotate the object on any axis.



- **Ctrl+Shift**+drag (Mac users **Ctrl+Shift**+drag) to constrain the rotation to one axis.



Transforming from the Node Properties Panel

The transform handles are a convenient way to move objects around in the 3D workspace, but when you want more precision, you should enter values directly into the object's node panel. The panel also includes transform and rotation order options, which are not available within the 3D Viewer.

The following assumes you've already loaded the object's parameters into the Properties Bin.

To set transformation options:

- From the **transform order** list, select an option to define the order by which transformations are executed (**s** signifies scale, **r**, rotation; and **t**, translation).
- From the **rotation order** list, select an option to define the axial order by which rotations are executed.

To transform an object from its panel:

- To move the object along one or more axes, increment or decrement the **translate x**, **y**, and **z** fields.
- To rotate the object, increment or decrement the **rotate x**, **y**, and **z** fields.
- To scale the object on all axes simultaneously, increment or decrement the **uniform scale** field.
- To scale the object asymmetrically (on x, y, or z), increment or decrement the **scale x**, **y**, and **z** fields.
- To skew the object (warp it by rotating its local axes), increment or decrement the **skew x**, **y**, and **z** fields to rotate the corresponding axis (and associated object vertices).

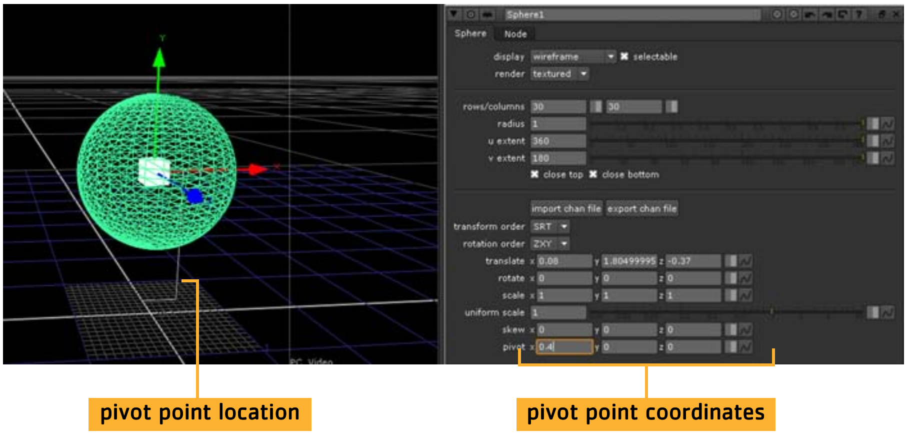
Transformations and the Pivot Point

When you make changes to an object's position, scaling and rotation, these occur from the location of the object's origin point or *pivot*. By default, the pivot point is located at the intersection of the object's local axes.

You can offset the pivot point and move it anywhere you like—you can even move it outside of the object. Subsequent local transformations will then occur relative to the new pivot point location.

To move the pivot point:

1. Double-click on the object node to display its parameters.
2. Change the values of the **pivot x**, **y**, and **z** fields to move the local axis in any direction.



display wireframe ✖ selectable
render textured
rows/columns 30 30
radius 1
u extent 360
v extent 180
✖ close top ✖ close bottom
transform order SRT
rotation order ZXY
translate x 0.08 y 1.80499995 z -0.37
rotate x 0 y 0 z 0
scale x 1 y 1 z 1
uniform scale 1
skew x 0 y 0 z 0
pivot x 0.4 y 0 z 0

pivot point location

pivot point coordinates

Figure 15.22: Pivot point location and coordinates.

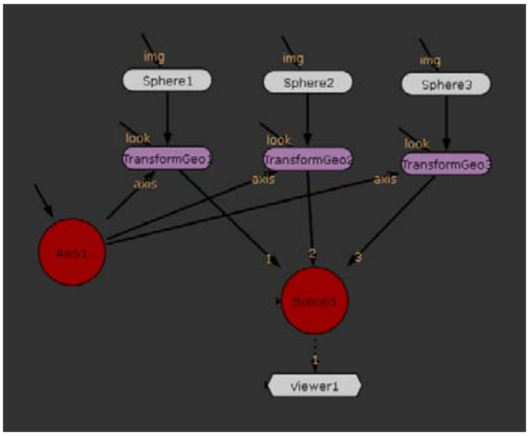
The object's graphical overlay points to the location of the pivot point with a line. All subsequent local transformations occur relative to this pivot point.

Once you've defined the location of an object's pivot point, you can use the object's transform parameters to translate, rotate, scale, and skew the object relative to the pivot point.

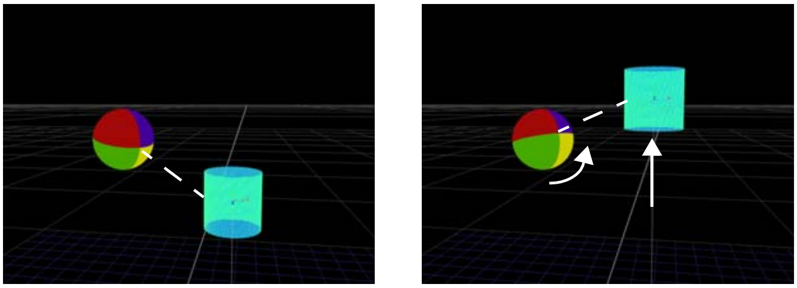
Using the TransformGeo Node

The TransformGeo node allows you to move, rotate, scale, and perform other transformations on several objects merged together with a MergeGeo node. It also lets you connect geometry objects to an Axis node. By doing so, you can move all the connected objects together by using the Axis transformation controls. All you need to do is insert a TransformGeo after each geometry object, connect the Axis node to the TransformGeo nodes'

axis input, and adjust the transform controls of the Axis node. For more information, see “Parenting to Axis Objects” on page 441.


img
Sphere1
img
Sphere2
img
Sphere3
look
TransformGeo1
look
TransformGeo2
look
TransformGeo3
axis
axis
axis
Axis1
1
2
3
Scene1
1
Viewer1

Another use of the TransformGeo node is to have the rotation of one object depend on the position of another so that the first object is always rotated to face or “look at” the second one. For example, you can have a sphere object always facing a cylinder object, regardless of the cylinder’s position. When the cylinder is moved to a new position, the sphere is automatically rotated to face it.



To have one 3D object always face another:

1. Select the 3D object node (for example, a sphere) that you want to face another object.
2. Choose **3D** > **Modify** > **TransformGeo** to insert a TransformGeo node.
3. Select the object you want the first object to face (for example, a cylinder), and insert a TransformGeo after this node, too.
4. Connect the first TransformGeo node into the **look** input of the second TransformGeo node.

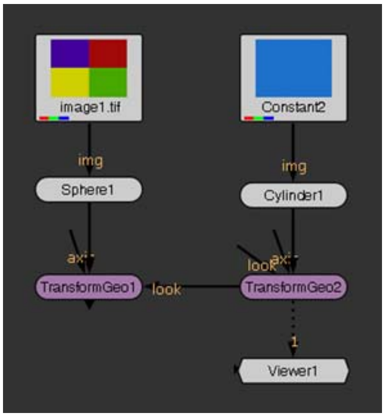
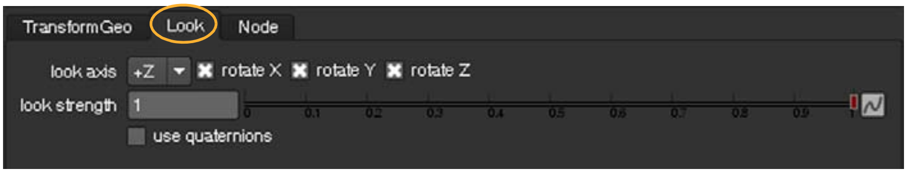


image1.tif
Constant2
img
img
Sphere1
Cylinder1
axis
look
axis
TransformGeo1
look
TransformGeo2
1
Viewer1

Open the controls of the first TransformGeo node and go to the **Look** tab.

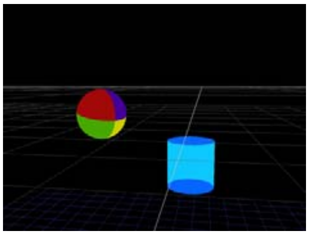

TransformGeo
Look
Node
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

look axis +Z ☒ rotate X ☒ rotate Y ☒ rotate Z

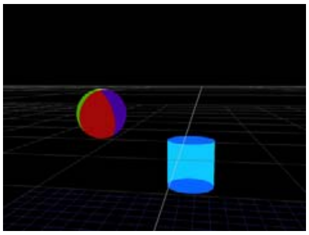
look strength 1

☐ use quaternions

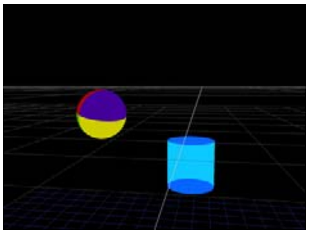
5. From the **look axis** pulldown menu, select the axis around which the object will be rotated to face the other object:



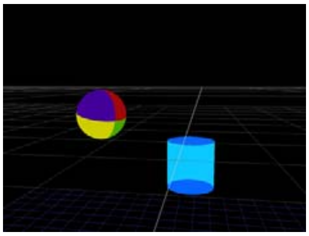
+Z



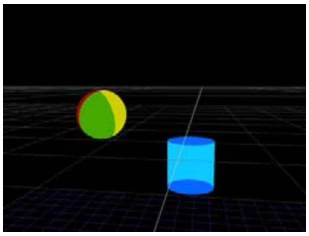
+Y



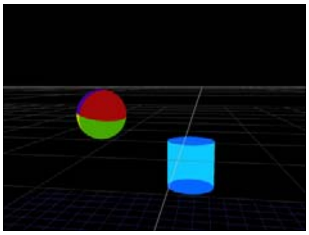
+X



-Z



-Y



-X

6. Use the **rotate X**, **rotate Y**, and **rotate Z** check boxes to select the axes the object will be rotated around. For the first object to truly face the second, you need to check all three check boxes.

7. Adjust the **look strength** slider to define the extend of the rotation. The smaller the value, the less the object is rotated. Setting the value to 0 produces no rotation.
8. If you want to use an alternate scheme to calculate the rotation, check **use quaternions**. This may be useful for smoothing out erratic rotations along the selected **look axis**.

If you now adjust the second TransformGeo node's transform controls, you'll notice that the first object automatically rotates to face the second object. For more information on how to adjust the transform controls, see "Using the Transform Handles" on page 458 and "Transforming from the Node Properties Panel" on page 459.

Modifying Object Shapes

Many nodes under the Modify menu let you modify the shape of an object as a whole. Modifying only selected portions of an object is currently not supported.

You can modify 3D objects using lookup curves, power functions, images, a Perlin noise function, a distortion function, and a trilinear interpolation.

Modifying Objects Using Lookup Curves

The CrosstalkGeo and LookupGeo nodes offer you direct global control over each of the vertex x, y, and z values respectively. You can, for example, only modify all the y values without touching the x and z values.

You change the different vertex values (x, y, or z) by modifying their associated 2D curves in lookup tables (LUTs). The x axis in the LUT represents the current vertex value, and the y axis the new vertex value.

By default, the curve is a diagonal line where all the points in the curve have the same value on the y axis (the new value) as they do on the x axis (the current value). Because both x and y values are the same, there is no change in the object's shape.

By modifying, for example, the CrosstalkGeo node's **y** LUT the following way, you can set some of the vertex y values of a sphere to 0 to squash its bottom half:

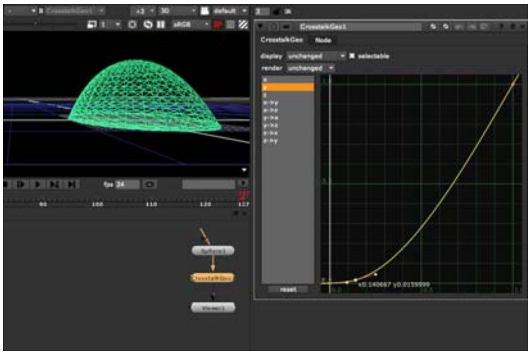


Figure 15.23: Modifying the CrosstalkGeo node's LUT.

With the CrosstalkGeo node, you can also use one of the vertex x, y, and z values to evaluate the lookup curve and then add the result to another vertex value. For example, you could modify the x->y curve, using the vertex x value to find the new value on the curve, and then add that to the vertex y value. This way, you can modulate the y values by another channel.

By default, these curves are horizontal lines at y=0. They produce no change, because the value added to the vertex (the new value on the y axis) is 0.

To modify objects using lookup curves:

1. Select **3D** > **Modify** > **CrosstalkGeo** or **LookupGeo** to insert a CrosstalkGeo or LookupGeo node anywhere after the 3D object you want to modify.
2. Attach a Viewer to the node to see your changes.
3. In the node's controls, use the **display** pulldown menu to select how you want to view your object in the Viewer while making changes to it.
4. From the list on the left, select the curve you want to modify. For example, you'd select **z** to only modify the vertex z values.

   In the case of the CrosstalkGeo node, you can also select **y->x**, for example, to use the vertex y value to evaluate the curve and add the result to the vertex x value.
5. Adjust the curve as necessary. To insert points on the curve, **Ctrl/Cmd**+**Alt**+click on the curve.

Modifying Objects Using a Power Function

The LogGeo node lets you modify the shape of your 3D objects using a power function. Using this node, you can raise each of the vertex x, y, and z values to a power (X^x, Y^y, Z^z). This can have a different effect depending on whether you are dealing with negative or positive values.

To modify objects using a power function:

1. Select **3D** > **Modify** > **LogGeo** to insert a LogGeo node anywhere after the 3D object you want to modify.
2. Attach a Viewer to the node to see your changes.
3. In the node's controls, use the **display** pulldown menu to select how you want to view your object in the Viewer while making changes to it. See "Object Display Properties" on page 456.
4. Check **swap**. This swaps the values and the powers they are raised to around (for example, changes 5^7 into 7^5).
5. In the **log x**, **y**, and **z** fields, enter the power you want to raise the respective vertex values to. For example, if you want to raise the vertex z values to the power of 20, enter 20 in the **z** field.

   Alternatively, you can adjust your 3D object in the Viewer by dragging the white control point to a new location. You can find the control point just outside the object.
6. To clamp the negative x, y, and z values to 0.0, check **clamp black**. This option is only valid if you have checked **swap**.

Tip *If you set the **log x, y,** and **z** values to 1 and check **swap**, the LogGeo node produces no change in the incoming geometry. If you want to try out how the node works, this is a good place to start as you can then gradually adjust the values from there.*

The following images illustrate the effect of the LogGeo node on the default Nuke cylinder and sphere when **swap** is checked in the LogGeo controls. Notice that if these objects were not positioned in the default location (centered around 0,0,0), the results would be different.

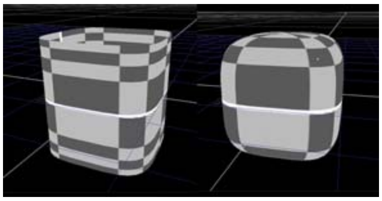


Figure 15.24: The LogGeo node applied to the default cylinder and sphere: Log x, y and z set to 0.5.

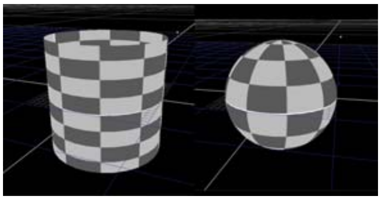


Figure 15.25: The LogGeo node applied to the default cylinder and sphere: Log x , y, and z set to 1 (no change).

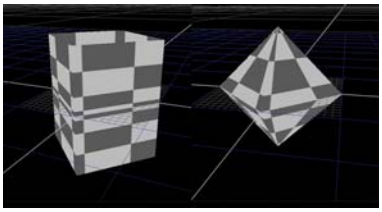


Figure 15.26: The LogGeo node applied to the default cylinder and sphere: Log x, y, and z set to 2.

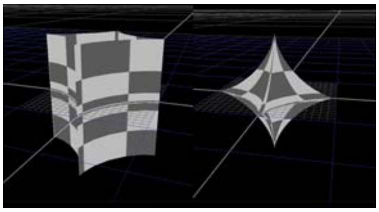


Figure 15.27: The LogGeo node applied to the default cylinder and sphere: Log x, y, and z set to 2.

Modifying Objects Using an Image

With the DisplaceGeo node, you can modify geometry based on an image. When using the node, each vertex is displaced along its normal with a value corresponding to the image pixel the vertex's uv attribute points to. The higher the pixel value, the greater the displacement.

The following image illustrates the principle behind the DisplaceGeo node. A Card node is modified to resemble the pattern of a Checkerboard image.

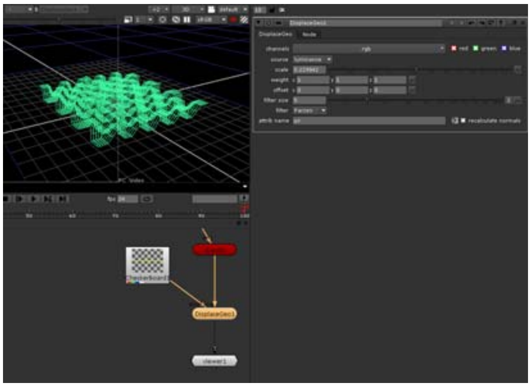


Figure 15.28: Using the DisplaceGeo node to modify geometry.

To modify objects using an image:

1. Select **3D** > **Modify** > **DisplaceGeo** to insert a DisplaceGeo node anywhere after the 3D object you want to modify.
2. Attach a Viewer to the node to see your changes.
3. In the node's controls, use the **display** pulldown menu to select how you want to view your object in the Viewer while making changes to it.
4. Read in your image map and connect it to the DisplaceGeo node's **displace** input.
5. Adjust the following controls:

- From the **channels** pulldown menu and check boxes, select the channels to use for the displacement value.
- From the **source** pulldown menu, select the source for the displace value. For example, if you selected **rgb** or **rgba** from the **channels** pulldown menu, you can use the red, green, or blue channel or the pixel luminance as the source.
- To define the scale of the displacement, adjust the **scale** slider. The higher the value, the bigger the displacement.
- To give x, y, and z different weightings, enter new weights the **weight** fields. By default, each weighting is set to 1. If you don't want to make changes to a value, set its weight to 0.
- To offset x, y, and z values, enter the value by which you want to offset them in the **offset** fields. For example, if you enter 0.5 in the y offset field, 0.5 is added to the y value.
- To change the size of the filtering applied to the image before the displacement, adjust the **filter size** slider.
- To select the filtering algorithm applied to the image before the displacement, select an algorithm from the **filter** pulldown menu. For more information, see "Choosing a Filtering Algorithm" on page 227.
- To change the name of the attribute that's used as the vertex's UV coordinates to find the image pixel, enter a name in the **attrib name** field.
- Usually, the normals aren't correct after the vertices have been moved. To recalculate them after the displacement, check **recalculate normals**.

Modifying Objects Using a Perlin Noise Function

The ProcGeo node lets you modify your 3D objects using a Perlin noise function that creates *seemingly* random noise. For example, you could use the ProcGeo node to generate animated noise for rippling waves or clouds, or to create a terrain from a flat card, like in the following image:

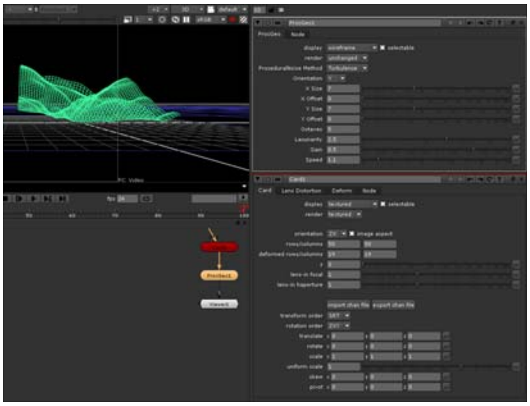


Figure 15.29: Using the ProcGeo node to create a terrain from a card object.

You can select the type of noise and control its look in the ProcGeo node's parameters.

To modify objects using a Perlin noise function:

1. Select **3D** > **Modify** > **ProceduralNoise** to insert a ProcGeo node anywhere after the 3D object you want to modify.
2. Attach a Viewer to the node to see your changes.
3. In the node's controls, use the **display** pulldown menu to select how you want to view your object in the Viewer while making changes to it.
4. From the **ProceduralNoise Method** pulldown menu, select the type of noise you want to use: **Turbulence** or **fBm** (Fractal Brownian Motion).
5. To select whether to modify the x, y, or z values or all of them, use the **Orientation** pulldown menu.
6. To change the look of the noise, adjust the rest of the parameters. For example, to control the amount of detail of the noise, adjust **Octaves**.

Modifying Objects Using a Distortion Function

The RadialDistort node is a non-linear transformation of the vertices along directions from the object center, giving either a barrel or pin-cushion distortion. In the following image, two cylinders have been distorted using the RadialDistort node.

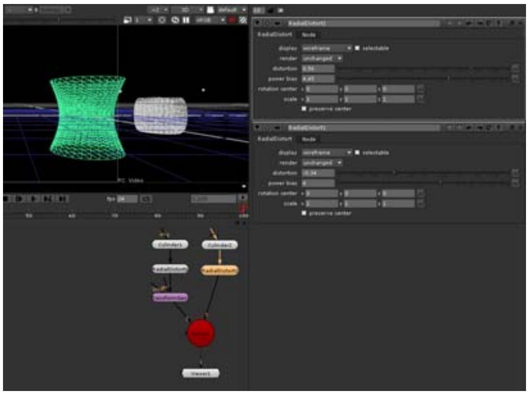


Figure 15.30: Barrel and pin-cushion distortions.

To modify objects using a distortion function:

1. Select **3D** > **Modify** > **Radial Distort** to insert a RadialDistort node anywhere after the 3D object you want to modify.
2. Attach a Viewer to the node to see your changes.
3. In the node's controls, use the **display** pulldown menu to select how you want to view your object in the Viewer while making changes to it.
4. To select whether the distortion is a barrel or pin-cushion, adjust the **distortion** slider. Values below 0 produce a barrel distortion, whereas values above 0 produce a pin-cushion distortion. If you set the value to 0, the 3D object is not distorted.
5. To control the magnitude of the distortion, adjust the **power bias** slider. The higher the value, the more distorted the object becomes.
6. To move the center point of the distortion, enter new coordinates in the **rotation center** fields.
7. To control the amount of distortion in each of the x, y, or z directions, adjust the values in the **scale** fields.
8. To keep the object's center in its original place in the 3D space, check **preserve center**.

Modifying Objects Using a Trilinear Interpolation

With the Trilinear node, you can warp the object as a whole by using a trilinear interpolation to warp the object's bounding box. For example, you can use this node to create animated object deformations, such as the squish/squash of a bouncing ball.

To modify objects using a trilinear interpolation:

1. Select **3D** > **Modify** > **Trilinear** to insert a Trilinear node anywhere after the 3D object you want to modify.

2. Attach a Viewer to the node to see your changes.
3. In the node's controls, use the **display** pulldown menu to select how you want to view your object in the Viewer while making changes to it.
4. To move each corner of the bounding box, enter new coordinates in the **p0**, **p1**, **p2**...**p7** fields. To cancel your changes and reset the box, select **reset shape to input**.
5. To not use the object's bounding box but define a box yourself, go to the **Source box** tab and uncheck **use incoming bounding box**. Adjust the **src0** and **scr1** coordinates define the box. To change the color of the box, click the **box** button.

Lighting

The nodes under the Lights menu let you control the lighting in your scene. Using these nodes, you can bring objects out or push them back, create an illusion of depth, simulate the conditions in the real world, or simply alter the feeling of the scene. You can also control the way your lights cast shadows.

Nuke features four types of light you can use in your 3D scenes: direct light, point light, spot light, and environment light. You can add these using the DirectLight, Point, Spotlight, and Environment nodes.

In addition to the nodes mentioned above, there is a Light node, which lets you read in lights from FBX files (for more information, see "Importing Channel Files, Cameras, Lights, Transforms, and Meshes from Other Applications" on page 483).

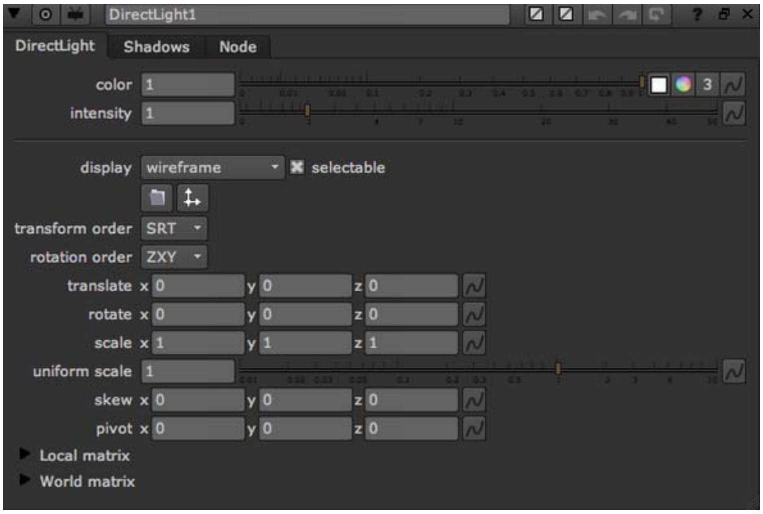
The Light node also includes the DirectLight, Point, and Spotlight nodes, so you can set it to act as any of these three nodes. Simply insert a Light node (select **3D** > **Lights** > **Light**) and choose the light type you want to use. The advantage of using a Light node in this way is that if you want to change the light type later, you can do so without setting up a new node. For example, you might insert a direct light, but then realize that what you actually need is a spot light. If you inserted the direct light using a DirectLight node, you need to delete this node and insert a SpotLight node instead. However, if you inserted the direct light using a Light node, you can simply change the light **type** from **directional** to **spot** in the Light controls.

Direct Light

A direct light is a light that emits parallel light in one direction. It appears to illuminate all objects with equal intensity, as if it was coming from a far away source. Being at an infinite distance from the objects, direct light has orientation, but no position. A real world example of a direct light is the sun. You can use direct light to simulate sunlight and moonlight, for example.

To add a direct light:

1. Select **3D** > **Lights** > **Direct** to insert a DirectLight node in your script.
2. Connect the DirectLight node to the Scene node.
3. In the DirectLight node's controls, adjust the following:
    - Drag the **color** slider to change the light color.
    - Drag the **intensity** slider to change the brightness of the light.
    - To control the direction of the light, enter values in the **rotate** fields.



color 1
intensity 1
display wireframe ☒ selectable
transform order SRT
rotation order ZXY
translate x 0 y 0 z 0
rotate x 0 y 0 z 0
scale x 1 y 1 z 1
uniform scale 1
skew x 0 y 0 z 0
pivot x 0 y 0 z 0

Figure 15.31: Direct light controls.

- To adjust the settings for shadows, change values for the controls on the **Shadows** tab. For more information on these controls, see "To adjust the Shadows controls" on page 476.

Point Light

A point light is a point in 3D space that emits light in every direction. A real world example of a point light is a light bulb. You can use point light to simulate light bulbs, lamps, and candles, for example.

To add a point light:

1. Select **3D** > **Lights** > **Point** to insert a Point node in your script.
2. Connect the Point node to the Scene node.

3. In the Point node's controls, adjust the following:
   - Drag the **color** slider to change the light color.
   - Drag the **intensity** slider to change the brightness of the light.
   - To control how much light the object gets from the light source (based on the distance between the object and the light source), use the **falloff type** menu. A **Linear** type diminishes the light at a fixed rate as it travels from the object, whereas **Quadratic** and **Cubic** types diminish the light at an exponential rate. If you select **No Falloff**, the distance between the light source and the object does not affect the lighting.
   - To control the position of the light in the 3D space, enter values in the **translate** fields.
   - To adjust the settings for shadows, change values for the controls on the Shadows tab. Note however that The Point light doesn't cast shadows if you're using ScanlineRender. For more information on these controls, see "To adjust the Shadows controls" on page 476.

Spot Light

A spot light is a point in 3D space that emits a cone-shaped light in a given direction. A real world example of a spot light is a desk lamp.

To add a spot light:

1. Select **3D** > **Lights** > **Spot** to insert a Spotlight node in your script.
2. In the node's controls, adjust the following:
   - Drag the **color** slider to change the light color.
   - Drag the **intensity** slider to change the brightness of the light.
   - Drag the **cone angle** slider to control the spread of the light (how wide or narrow the beam is) in degrees from 0 to 180.
   - Drag the **cone penumbra angle** slider to control the softness along the edge of the area of illumination. A negative value fades inward from the circle's edge. A positive value fades outward from the circle's edge. The **cone falloff** should be set to zero or a low value in order to see the softness. This feature is only visible in the rendered objects and not in the 3D OpenGL Viewer.
   - Drag the **cone falloff** slider to control how concentrated the light is (that is, how much the light diminishes from the center of the circular region out to the edge). The higher the value, the more focused the light becomes. The falloff is independent of the **falloff type**.
   - To control how much light the object gets from the light source (based on the distance between the object and the light source), use the **falloff type** menu. A **Linear** type diminishes the light at a fixed rate as it travels from the object, whereas **Quadratic** and **Cubic** types diminish

the light at an exponential rate. If you select **No Falloff**, the distance between the light source and the object does not affect the lighting.

- To control the direction of the light, enter values in the **rotate** fields.
- To control the position of the light in the 3D space, enter values in the **translate** fields.
- To adjust the settings for shadows, change values for the controls on the **Shadows** tab. For more information on these controls, see "To adjust the Shadows controls" on page 476.

Environment Light

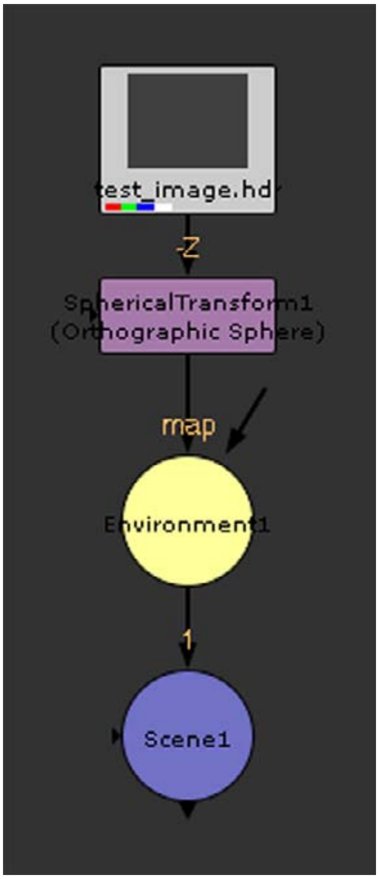
An environment light is a light that illuminates the objects using an image of light from a real-world environment. This image-based lighting is generated using High Dynamic Range Images (HDRI). When HDR images are created, several differently exposed images are combined to produce a single image of the surrounding environment. As a result, HDR images have a wide range of values between light and dark areas, and represent the lighting conditions of the real world more accurately.

To use environment light, you first need to shoot a real life environment as an HDR image. Using the SphericalTransform node, you then convert this image into a spherical mapped image. The sphere is used to surround the 3D objects, so that the mapped image color illuminates them.

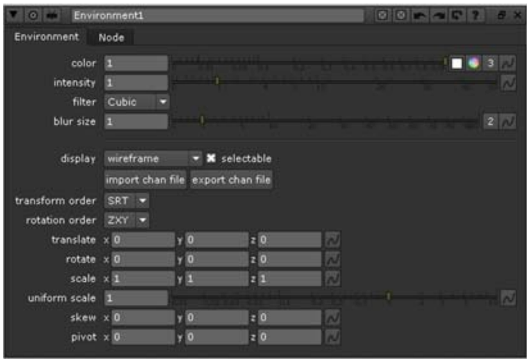
Environment light only works with shiny object materials that can reflect the mapped image. It results in a very realistic lighting that makes it easier to integrate the objects into the environment.

To add an environment light:

1. Read an HDR image of the environment into your script.
2. Select **Transform** > **SphericalTransform** to insert a SphericalTransform node after the HDR image. You use this node to convert the HDR image into a spherical mapped image. In the nodes controls, select the **Input Type** and the **Output Type** (in this case, **Sphere**).
3. Select **3D** > **Lights** > **Environment** to insert an Environment node in your script. Connect the SphericalTransform node to the Environment node's **map** input, and the Environment node to the Scene node.


test_image.hdr
-Z
SphericalTransform1
(Orthographic Sphere)
map
Environment1
1
Scene1

4. In the Environment node's controls, adjust the following:
   - Drag the **color** slider to change the light color.
   - Drag the **intensity** slider to change the brightness of the light.
   - From the **filter** pulldown menu, select a filtering algorithm for the map image. For more information, see "Choosing a Filtering Algorithm" on page 227.
   - To change the blur size of the map image, adjust the **blur size** slider.



color 1
intensity 1
filter Cubic
blur size 1
display wireframe ☒ selectable

transform order SRT
rotation order ZXY
translate x 0 y 0 z 0
rotate x 0 y 0 z 0
scale x 1 y 1 z 1
uniform scale 1
skew x 0 y 0 z 0
pivot x 0 y 0 z 0

Figure 15.32: Environment light controls.

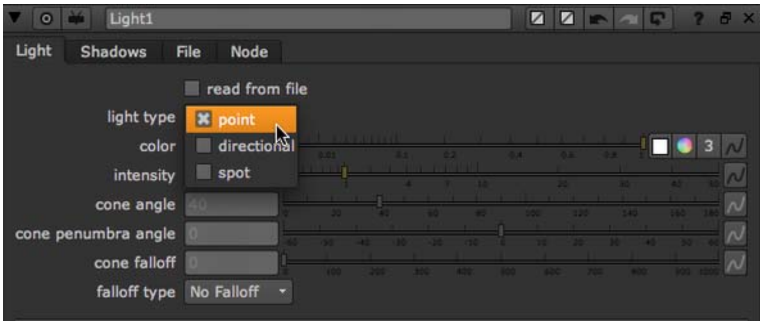
The Light Node

You can use the Light node to add a direct light, point light, or spot light into your script.

The node can also be used to import lights from FBX files. This is described later, under “Importing Channel Files, Cameras, Lights, Transforms, and Meshes from Other Applications” on page 483.

To add a direct, point, or spot light:

1. Select **3D** > **Lights** > **Light** to insert a Light node into your script.
2. In the Light controls, select the **light type** you want to use: **point**, **directional**, or **spot**. The controls are enabled and disabled according to the light type you choose. For example, if you chose directional light, you get the same controls that appear on the DirectLight node.



☐ read from file
light type ☒ point ☐ directional ☐ spot
color
intensity
cone angle 40
cone penumbra angle 0
cone falloff 0
falloff type No Falloff

3. Adjust the controls as necessary. For information on the functions of the controls, refer to the following:
   - If you selected point as the light type, see “Point Light” on page 471.

- If you selected directional as the light type, see "Direct Light" on page 470.
- If you selected spot as the light type, see "Spot Light" on page 472.

To adjust the Shadows controls

The Light, Point, Direct and Spot nodes all have controls for shadows that you can use to adjust how the lights cast shadows in your 3D scene. The method used to create shadows varies between different render nodes; ScanlineRender for example uses depthmapping to create shadows whereas PRmanRender creates shadows through raytracing. Some shadows controls are only relevant to particular shadow creation method. On the **Shadows** tab, adjust the following controls:

- **cast shadows** - check to set this light to cast shadows.
- **samples** - set the number of samples for the light when generating soft shadows.
- **sample width** - set the size of the light for soft shadows. This is only relevant if your shadows will be generated through raytracing.
- **depthmap offset** - set the bias for the shadowmap. Increase this value if self shadowing artefacts appear in the image. This control isn't relevant if your shadows are generated through raytracing.
- **jitter scale** - Amount of jitter used when doing PCF for soft shadows. A larger value results in softer shadows. This control isn't relevant if your shadows are generated through raytracing.
- **depthmap resolution** - set the resolution of the depthmap. Larger values will result in a less crunchy edge, but will require more time to process. Crunchy edges can also be fixed by increasing the number of samples instead of increasing the depthmap resolution. This control isn't relevant if your shadows are generated through raytracing.
- **output mask** - select the channel you want to output the shadowmap into. This can be enabled even if the **cast shadows** box is disabled.

Tip *To generate accurate shadows from a Direct light, view the scene through the light (using the Viewer's camera menu, just like a camera) and adjust the Direct light's* ***scale*** *control so that the part of the scene that should cast shadows fits within the view. This will ensure that none of the shadow-casting geometry is missed by the depth map.*

Manipulating Object Normals

Object normals are vectors that are perpendicular to the surface. They are used in lighting calculations to determine how the light should bounce off a surface at any particular point. By manipulating them, you can control the diffuse and specular light contributions.

To manipulate object normals:

1. Select **3D** > **Modify** > **Normals** to insert a Normals node anywhere after the 3D object whose lighting you want to adjust.
2. Connect a Camera, Axis, or light node to the Normals node's **lookat** input.
3. In the Normals controls, open the **action** pulldown menu and select:
    - **unchanged** to make no changes.
    - **set** to assign the normals value to the normal x, y, and z fields.
    - **build** to rebuild each normal based on the surrounding vertices. Adjust the **threshold angle** slider to determine the break angle where two faces no longer constitute a smooth surface. An angle of 0 means all faces are flat, whereas 180 means all faces are smooth. A good average setting is 60.
    - **lookat** to point all normals towards the Normals node's **lookat** input.
    - **delete** to remove the named attribute from the object. For example, if you remove the N attribute, the object has no normals.

Working with Cameras

Nuke lets you add multiple cameras to a scene, with each providing a unique perspective. You can also setup cameras that project 2D images onto geometry in your scene.

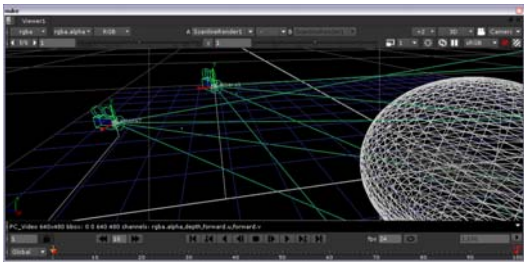


Figure 15.33: Cameras in the scene.

To add a camera:

1. Click **3D > Camera** to insert a Camera node.
2. To setup the rendering camera, drag a connector from the new Camera node to the ScanlineRender node.

   or

   To setup an additional scene camera for viewing, drag a connector from the new Camera node to the Scene node.

To "see" the scene through a particular camera, you need to select the camera and press **H**. Make sure you're in the 3D perspective view (press **V** before **H**) before trying this; it doesn't work when you're looking at your scene through an orthographic view.

To edit a camera's lens characteristics:

1. If necessary, double-click on the Camera node to display its parameters.
2. Click the **Projection** tab.
3. Drag the **focal length** slider to adjust the camera's level of magnification.
4. Drag the **near** slide to edit the position of the camera's forward clipping plane. Objects in front of this plane will not be rendered or displayed.
5. Drag the **far** slider to edit the position of the camera's rearward clipping plane. Objects in behind this plane will not be rendered or displayed.
6. Increment the **window translate u** (horizontal axis) and **v** (vertical axis) sliders to translate the camera's output along either axis.
7. Increment the **window scale u** (horizontal axis) and **v** (vertical axis) sliders to scale the camera's output on either axis.
8. Drag the **window roll** slider to rotate the camera's output on z.

Projection Cameras

In addition to viewing and rendering a 3D scene, cameras can also project a 2D still image or image sequence onto geometry in the scene. This is similar to the front-projection systems used in practical photography, where a background image or other element is projected onto the stage and photographed with other elements.

In Nuke, a projection camera can receive camera data tracked from the original shot—or another shot—to setup a projection that is match-moved to another source.

This setup requires these nodes: a projection camera, a Scene node, a Project3D node, a geometry object node (what you'll be projecting onto), and a 2D node with the image that you want to project.

First a Little Math...

When you create a projection camera, you need to gather some information and do a few small calculations to make sure the projection works. Here are the bits of information you need:

- Focal length of the lens that photographed the projection image.
- Resolution of scanned image.
- Scanner pitch of the film scanning device.

After you have this information, you need to do these calculations to get the horizontal and vertical aperture settings for the projection setup:

```
horiz. res. / scanner pitch = horizontal aperture
vertical res. / scanner pitch = vertical aperture
```

So, for example, if your image resolution is 720 x 486 and the scanner pitch is 20, then these are the results:

```
720 / 20 = horizontal aperture = 36
486 / 20 = vertical aperture = 24.3
```

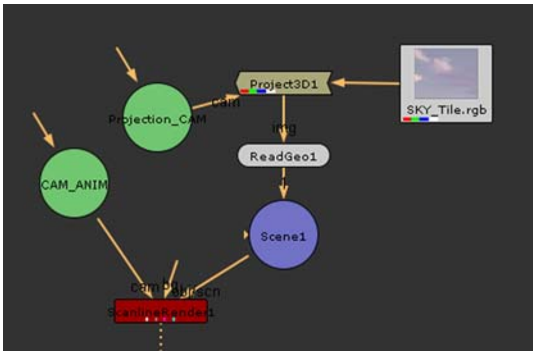
Generally, for most professional projects, you can get the lens focal length from the camera report for the shot(s). If that is not available, you may be able to extrapolate lens information by running the shot through a 3D tracking application, such as Boujou, Syntheyes, or RealViz.

Setting Up the Projection Camera Script

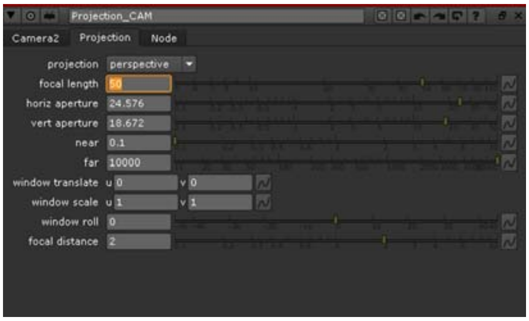
Once you have the horizontal and vertical aperture and the lens focal length for the image you want to project, you can complete the projection camera setup.

To add a projection camera:

1. Choose **3D > Camera** to add a new camera to your script and rename the node to identify it as a projection camera.
2. Choose **3D > Shader > Project3D** to add a Project3D node to the script.
3. Connect the 2D image (i.e., Read node) to the Project3D node.
4. Connect the projection camera to the Project3D node.
5. Connect the Project3D node to the geometry node that should receive the 3D projection.
6. Double-click the projection camera node to load its parameters.



7. Click the Projection tab in the camera's panel and then enter the information you gathered for **focal length**, **horiz aperture**, and **vert aperture**.



projection perspective
focal length 50
horiz aperture 24.576
vert aperture 18.672
near 0.1
far 10000
window translate u 0 v 0
window scale u 1 v 1
window roll 0
focal distance 2

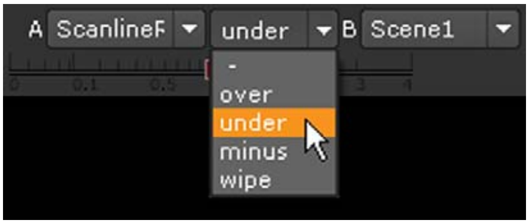
Figure 15.34: Entering projection camera settings.

When you are finished, view the 3D scene to check the placement of the projection. The next section explains how to preview the 2D and 3D elements together, to check the results of the composite.

To view a 3D scene over a 2D background image:

1. Select the Scene node and press **1** to display its output to the Viewer.
2. If necessary, press **Tab** to toggle the Viewer to 3D mode.
3. Select the rendering camera object or node and press **H** to look through it.
4. Select the node with the 2D image you want to see in the Viewer, and then press **Shift+2**.

   The Shift+2 keystroke connects the image to the Viewer (assigning the next available connection, number 2), and also sets up the compare wipe.
5. Choose the desired option from the Viewer composite list (i.e., **-** (none), **over**, **under**, **minus, wipe**).



This last step superimposes the two elements in the Viewer. The crosshair (shown below in Figure 15.35) is the control that lets you adjust the location and angle of the wipe for the comparison.

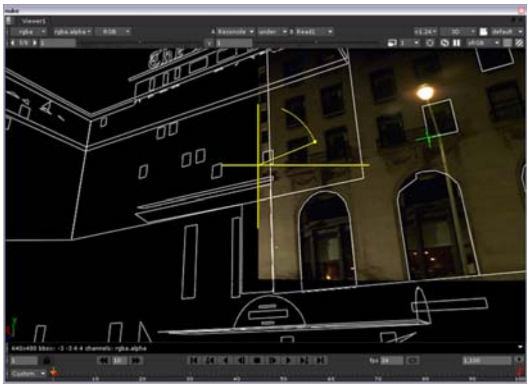


Figure 15.35: Comparing a 3D scene over a 2D image.

Adding Motion Blur to the 3D Scene

To create more realism for a 3D scene, you'll want to add motion blur to it based on the movement of your 3D camera. This can be done in two ways:

- If you have moving objects in your 3D scene or the camera movement over the shutter time is non-linear, adjust the **samples** value in the ScanlineRender node's parameters. The image is sampled multiple times over the shutter period. This way is the most accurate, but also the slowest, because the full rendering calculation is done multiple times for each pixel.
- If your 3D scene is static or nearly so and the camera movement over the shutter time is nearly linear, add the MotionBlur3D and VectorBlur nodes after the ScanlineRender node in your script. This way is faster to render.

To add motion blur for a scene with moving objects or non-linear camera movement during the shutter period:

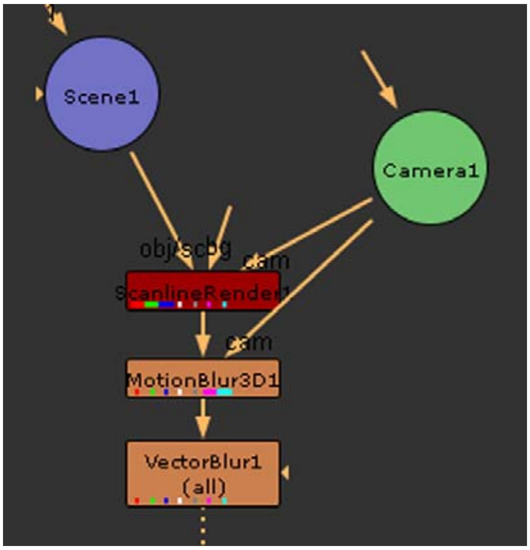
1. In the ScanlineRender node's controls, go to the **MultiSample** tab.
2. Increase the **sample** value to sample the image multiple times over the shutter period.

Tip *By increasing both the **sample** and **focus diameter** values in the ScanlineRender node's controls, you can also use the multisampling to simulate depth of field.*

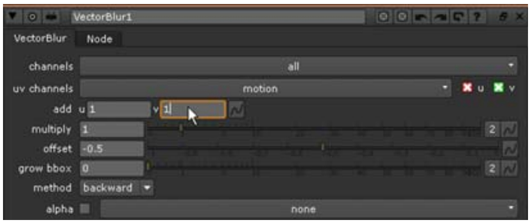
To add motion blur for a static scene with approximately linear camera movement during the shutter period:

1. Select the **ScanlineRender** node.

2. Choose **Filter** > **Motion Blur 3D** to insert this node and connect it to the scanline renderer.
3. Connect the rendering camera to the MotionBlur3D node.
4. Choose **Filter** > **Vector Blur** to insert and connect this node to the MotionBlur3D node.


Scene1
Camera1
obj/scn
bg
cam
ScanlineRender1
cam
MotionBlur3D1
VectorBlur1
(all)

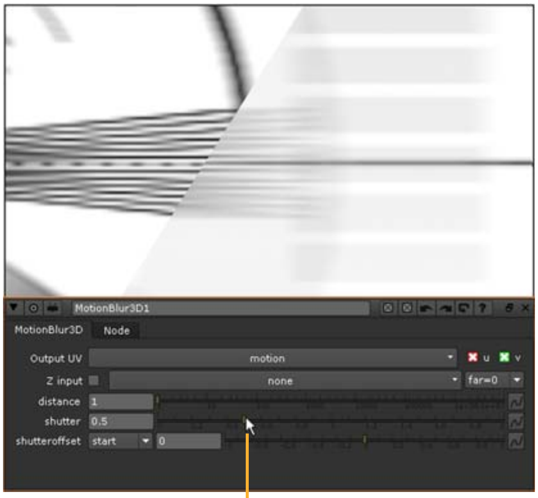
5. In the VectorBlur settings, select the **motion** layer from the **uv channels** list.
6. For the **add** settings, enter 1 for both **u** and **v**.


VectorBlur1
VectorBlur
Node

channels: all
uv channels: motion ☒ u ☒ v
add u 1 v 1
multiply 1
offset -0.5
grow bbox 0
method backward
alpha ☐ none

Note *For anamorphic footage, the add u value should be 2 and the add v value should be 1.*

7. To adjust the length of the blur, adjust the **Shutter** setting in the MotionBlur3D properties panel.



Output UV motion u v
Z input none far=0
distance 1
shutter 0.5
shutteroffset start 0

Increase or decrease the shutter value

Importing Channel Files, Cameras, Lights, Transforms, and Meshes from Other Applications

Sometimes, you may need to import files or objects created in 3D applications, such as Maya or Boujou. Depending on what you want to import and from where, there are different ways of doing the import:

- To apply motion data calculated with 3D tracking software to cameras or objects, you need to import channel files. For more information, see "Applying Tracks to an Object" below.

  You can also use channel files to import cameras created in other applications into Nuke. However, as the chan file format is not a standard file format, you may need a file format converter to export chan files from other applications.
- To import cameras, lights, transforms, or meshes from other applications, you can use FBX files. FBX is a standard file format many applications can export to. FBX files contain 3D scenes from which you can import cameras, lights, transforms, and meshes into Nuke. For more information, see "Working with FBX Files" on page 484.
- To import cameras from Boujou, you can use the import_boujou.tcl script shipped with Nuke. For more information, see "Importing Cameras from Boujou" on page 490.

Applying Tracks to an Object

Nuke can import channel files and apply the motion data to the transformation parameters of any camera or object. The most common purpose for this is to simulate a practical camera move or move objects along a defined path.

Channel files contain a set of cartesian coordinates for every frame of animation in a given shot. This information is calculated by 3D tracking software, such as 3D-Equalizer, Maya, or Boujou, and then exported as channel files.

To apply a channel file to an object:

1. Double-click on an object or camera node to display its parameters.
2. Click **import chan file**. The file navigation dialog appears.
3. Navigate to the channel file, then click **OK**.
4. Nuke reads in the channel data and displays a status message about the number of data frames imported. You'll also notice the object's translation parameters turn green to indicate these parameters are now controlled by animation data. Scrub the Viewer and you'll notice the object or camera now moves according to the transformation data imported from the channel file.

Note *You can use the **export chan file** button to export as a chan file any animated translation parameters which you've applied to given object. This is a useful method of sharing setups between artists.*

Working with FBX Files

FBX is a standard 3D file format that gives you access to 3D scenes created in other applications supporting the same format. What you generally have in an FBX file is an entire 3D scene containing cameras, lights, meshes, non-uniform rational B-spline (NURBS) curves, transformation, materials, and so on. From this scene, you can extract cameras, lights, transforms, and meshes into Nuke. This way, you can, for example, create a camera in Maya, export it in an FBX file, and use the same camera again in Nuke.

To extract cameras, lights, transforms, and meshes into Nuke from FBX files created in other applications, use the following nodes:

- for cameras, the Camera node
- for lights, the Light node
- for transforms, the Axis node
- for meshes (or NURBS curves/patch surfaces converted to meshes), the ReadGeo node.

All these nodes include similar controls for handling FBX files, as you will notice from their descriptions below.

Note *The FBX SDK reads FBX files produced by Autodesk MotionBuilder versions 5.5 and later. The FBX SDK writes FBX files compatible with MotionBuilder (version 5.5 and later) and earlier versions of the Autodesk FBX SDK (6.0, 7.0, 2005.12, and later).*

Tip *If you have trouble with FBX files, it may be because they were written with an older version of FBX. If they load very slowly, it is also possible that they are ASCII rather than binary. To get around these problems, you can use the FBX converter on the Autodesk web site. It converts between various different formats, including older FBX versions, ASCII, and binary, and is available on Windows, Mac OS X, and Linux.*

To download the FBX converter:

1. Go to http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=10775855 .

2. Scroll down to ***FBX Converter*** *and click on one of the links to start the download.*

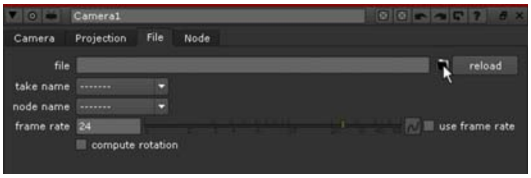
Importing cameras from an FBX file

The Camera node lets you read in the standard fbx cameras (Producer Perspective, Producer Top, Producer Bottom, Producer Right, Producer Left, Producer Front, Producer Back) and any other cameras.

Using one Camera node, you can only import one camera from an FBX file. If you need to import several cameras, you need to use one Camera node per camera.

To import a camera from an FBX file:

1. Select **3D** > **Camera** to insert a Camera node in the place where you want to add the camera in your script.
2. In the Camera controls, check **read from file**. When this is checked, the controls on the **File** tab are enabled, and you can use them to read in a camera from an FBX file. Any controls whose values are read in from the FBX file are disabled. You can still view these values and use them in expressions but, as long as **read from file** is checked, you cannot modify them. Modifying the values in the FBX file, however, will affect the disabled values in the Camera controls, because these are reloaded from the FBX file every time the node is instantiated.
3. To read in a camera from an FBX file, click the folder icon on the **File** tab. Navigate to the FBX file and select **Open**.



file
take name --------
node name --------
frame rate 24
☐ use frame rate
☐ compute rotation

4. From the **take name** pulldown menu, choose the take you want to use from the FBX file. FBX files support multiple takes in one file. Usually, one of the takes is a default take with no animation.
5. From the **node name** pulldown menu, select the camera node you want to import from the FBX file.
6. In the **frame rate** field, define a frame rate (frames per second) to sample the animation curves. To use this rate rather than the one defined in the FBX file, check **use frame rate**.
7. To have the camera rotation values calculated using the look up vector and look at position, check **compute rotation**. If you don't check this, Nuke uses the rotation channel from the FBX file instead of computing a new one. The rotation values are always computed when there is a look at target.
8. If you want to modify the camera properties imported from the FBX file, uncheck **read from file** on the **Camera** tab and make the necessary modifications. As long as **read from file** is unchecked, your changes are kept.
9. To reload the camera properties from the FBX file, make sure **read from file** is checked and click the **reload** button on the **File** tab.

Importing lights from an FBX file

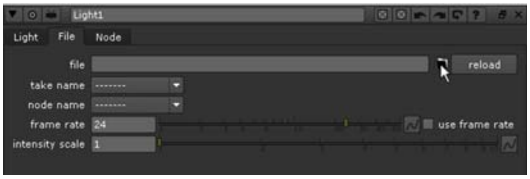
You can use the Light node to read in directional, point, and spot lights from FBX scene files (for more information on these three light types, refer to "Lighting" on page 470). One Light node only reads in one light. Therefore, if your FBX file contains three lights and you want to import all of them into Nuke, you need to use three Light nodes.

To import a light from an FBX file:

1. Select **3D** > **Lights** > **Light** to insert a Light node in the place where you want to add the light in your script.
2. In the Light controls, check **read from file**. This enables the controls on the **File** tab, allowing you to read in lights from an FBX file. It also disables all controls whose values will be filled from the FBX file. You can still view these values and use them in expressions, but you cannot

modify them, because they are read from the FBX file. Any changes you make in the FBX file will be reflected in these values of the Light node.

3. On the **File** tab, click the folder icon and browse to the FBX file that contains the light you want to use. Click **Open**.



file
take name -------
node name -------
frame rate 24 ☐ use frame rate
intensity scale 1

4. From FBXthe **take name** pulldown menu, select the take you want to use from the FBX file. FBX files support multiple takes in the same file. One of the takes is usually a default take without any animation.
5. From the **node name** pulldown menu, select the light node you want to import from the FBX file.
6. If you want to override the frame rate used in the FBX file to sample the animation curves, enter a new frame rate (frames per second) in the **frame rate** field. Check **use frame rate** to use the rate you entered (rather than the one in the FBX file).
7. To scale the intensity channel values read from the FBX file, adjust the **intensity scale** slider. If the light is too dark, increase this value.
8. If you want to modify the light properties imported from the FBX file, uncheck **read from file** on the **Light** tab and make the necessary modifications. As long as **read from file** is unchecked, your changes are kept.
9. To reload the light properties from the FBX file, make sure **read from file** is checked and click the **reload** button on the **File** tab.

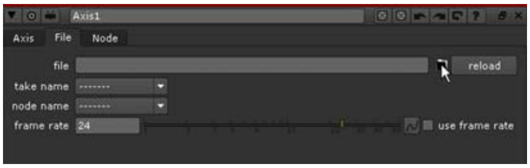
Importing transforms from an FBX File

The Axis node reads in transforms, markers and nulls (locators) from FBX files. You can use it to import one transform, marker, or null per Axis node.

To import a transform from an FBX file:

1. Select **3D** > **Axis** to insert an Axis node in your script. Connect the Axis node to a Scene node.
2. In the Axis controls, check **read from file**. This enables the controls on the **File** tab, allowing you to import transforms from an FBX file. It also disables controls whose values are filled in from the FBX file. As long as **read from file** is checked, you cannot modify these values. You can, however, view them and use them in expressions. The values are

reloaded from the FBX file every time the node is instantiated, so any changes you make in the FBX file's values will be reflected in the Axis controls.



3. On the **File** tab, click the folder icon to open the File Browser. Navigate to the FBX file that contains the transform you want to use. Click **Open**.
4. From the **take name** pulldown menu, select the take you want to use from the FBX file. FBX files support multiple takes, one of which is usually a default take with no animation.
5. From the **node name** pulldown menu, choose the transform, marker, or null you want to import from the FBX file.
6. If you do not want to use the frame rate from the FBX file for sampling the animation curves, in the **frame rate** field, enter a new value (frames per second). To override the frame rate defined in the FBX file and use the one you defined here, check **use frame rate**.
7. If you want to modify the transform properties imported from the FBX file, uncheck **read from file** on the **Axis** tab and make the necessary modifications. As long as **read from file** is unchecked, your changes are kept.
8. To reload the transform properties from the FBX file, make sure **read from file** is checked and click the **reload** button on the **File** tab.

Importing meshes from FBX files

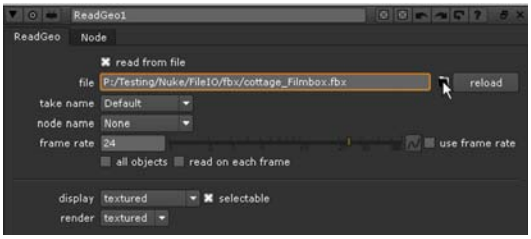
The ReadGeo node lets you import meshes (or NURBS curves/patch surfaces converted to meshes) from FBX files. Using one ReadGeo node, you can read in a single mesh or all the meshes in an FBX file.

The mesh's vertices, normals, UV's, and vertex colors are read on a per frame basis FBXor at frame 0. If there are any shape or cluster deformers, they are applied to the vertices. Materials or textures are not read in.

To import a mesh from an FBX file:

1. Select **3D** > **Geometry** > **ReadGeo** to insert a ReadGeo node into your script.

2. In the ReadGeo controls, click the folder icon next to the **file** field and navigate to the FBX file that contains the mesh you want to import. Click **Open**.



☒ read from file
file P:/Testing/Nuke/FileIO/fbx/cottage_Filmbox.fbx
take name Default
node name None
frame rate 24 ☐ use frame rate
☐ all objects ☐ read on each frame
display textured ☒ selectable
render textured

3. Make sure **read from file** is checked. This enables the file controls below. It also disables any controls whose values will be filled in from the FBX file. You can view these values and use them in expressions, but as long as **read from file** is checked, you cannot modify them. Any changes in the FBX file's values are reflected in the ReadGeo controls, however, because the values are reloaded from the FBX file every time the node is instantiated.
4. From the **take name** pulldown menu, choose the take you want to use. FBX files support multiple takes. Usually, one of them is a default take that contains no animation.
5. From the **node name** pulldown menu, select the mesh you want to import from the FBX file.
6. To adjust the frame rate used to sample the animation curves, enter a new value (frames per second) in the **frame rate** field. The frame rate you enter is only used if you check **use frame rate**. Otherwise, the frame rate from the FBX file is used.
7. If you want to import all the meshes in the FBX file rather than just one, check all objects. This overrides whatever you have selected under node name. If the objects are animated, check read on each frame. This will bake each object's transform into the mesh points and preserve the animation.
8. If you want to modify the transform properties imported from the FBX file, uncheck **read from file** and make the necessary modifications. As long as **read from file** is unchecked, your changes are kept.
9. To reload the transform properties from the FBX file, make sure **read from file** is checked and click the **reload** button.

Importing point clouds from FBX files

The ReadGeo node also lets you import point clouds from FBX files. A point cloud can be created using the CameraTracker node, for example.

To import a point cloud from an FBX file:

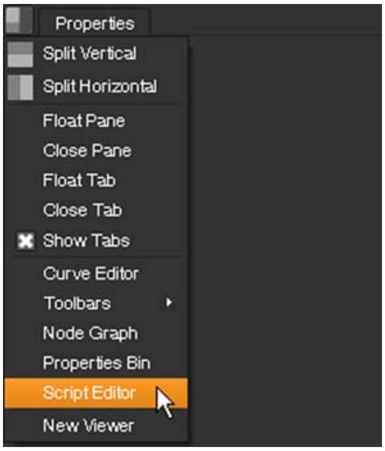
1. Select **3D** > **Geometry** > **ReadGeo** to insert a ReadGeo node into your script.
2. In the ReadGeo controls, click the folder icon next to the **file** field and navigate to the FBX file that contains the point cloud you want to import. Click **Open**.
3. Make sure **read from file** is checked. This enables the file controls below. It also disables any controls whose values will be filled in from the FBX file. You can view these values and use them in expressions, but as long as **read from file** is checked, you cannot modify them. Any changes in the FBX file's values are reflected in the ReadGeo controls, however, because the values are reloaded from the FBX file every time the node is instantiated.
4. From the **take name** pulldown menu, choose the take you want to use. FBX files support multiple takes. Usually, one of them is a default take that contains no animation.
5. In the **object type** drop-down, select **Point Cloud**.
6. To adjust the frame rate used to sample the animation curves, enter a new value (frames per second) in the **frame rate** field. The frame rate you enter is only used if you check **use frame rate**. Otherwise, the frame rate from the FBX file is used.
7. If you want to modify the transform properties imported from the FBX file, uncheck **read from file** and make the necessary modifications. As long as **read from file** is unchecked, your changes are kept.
8. To reload the transform properties from the FBX file, make sure **read from file** is checked and click the **reload** button.

Importing Cameras from Boujou

Nuke is shipped with a script called import_boujou.tcl, which lets you load in cameras created with Boujou.

To import a camera from Boujou:

1. Save the Boujou camera solve as a .txt file.
2. In Nuke, click on a content menu button and select Script Editor. The Script Editor opens in the pane whose content menu you used.


Properties
Split Vertical
Split Horizontal
Float Pane
Close Pane
Float Tab
Close Tab
Show Tabs
Curve Editor
Toolbars
Node Graph
Properties Bin
Script Editor
New Viewer

3. In the input pane of the Script Editor (that is, the lower pane), enter **nuke.tcl("import_boujou")**. Click the **Run the current script** button on the top of the Editor, or press **Ctrl+Return** (**Cmd+Return** on a Mac).
4. In the File Browser that opens, navigate to the .txt file you saved in step 1.

A Camera, a ScanlineRender, and a Group node are loaded into Nuke. The Group node contains cylinders to represent points from Boujou.

Tip *You can also open Nuke's Boujou Text File Browser by doing the following:*

*1. Press **x** on the Node Graph to open the Nuke script command dialog.*

*2. In the dialog, check **TCL** (if it's not already checked).*

*3. In the command field, enter **import_boujou**.*

*4. Click **OK**.*

These steps can be used to replace the first three steps in the above instructions.

Exporting Geometry, Cameras, Lights, Axes, or Point Clouds

You can also export geometry, cameras, light, axes and point clouds into an FBX file using the WriteGeo node.

1. Create a WriteGeo node and attach it to a Scene node.
2. In the WriteGeo controls, select **fbx** in the **file type** drop-down, and under **fbx Option**, check:
   - **geometries** - to write the scene geometries into the fbx file
   - **cameras** - to write the scene cameras to the fbx file
   - **lights** - to write the scene lights into the fbx file

  - **axes** - to write the scene axes into the fbx file
  - **point clouds** - to write the scene point clouds into the fbx file.
3. Check **ascii file format** if you don't want to write out a binary fbx file.
4. Check **animate mesh vertices** if you want the mesh vertices animated and keyframes created at every frame. The animated meshes use vertex point cache for the data. A directory with the **_fpc** appended to the file name is created to contain the point caches. By default the animated meshes check box is off so the point cache directory is not created.
5. Browse to the destination you want to save the FBX file in the **file** field and give it a name.
6. From the **file type** pulldown menu, select the file format for the rendered images. If you don't specify a file format, Nuke uses the extension in the file name to figure out the format.
7. Check the **limit to range** box if you want to disable the node when executed outside of a specified frame range. In the **frame range** fields, enter the range of frames you want to make executable with this Write node.
8. Click **Execute**.
9. If necessary, you can change the frame range you want to include in the file n the pop-up dialog and click **OK**.

A progress bar will display, and an FBX file is saved.

Rendering a 3D Scene

The 3D Viewer displays the scene using an OpenGL hardware render. When you build a scene, Nuke renders high-quality output from the perspective of the camera connected to the ScanlineRender node. The rendered 2D image is then passed along to the next node in the compositing tree, and you can use the result as an input to other nodes in the script.

To render out a scene:

1. Make sure the rendering camera is connected to the ScanlineRender node.
2. Toggle the Viewer back to 2D.
3. Connect the output of the ScanlineRender node to the appropriate 2D nodes in your script.

Adjusting the Render Parameters

You can affect the rendered output by adjusting the various controls of the ScanlineRender node.

You can, for example, select the projection mode to do have different renderings of the scene, or change the global ambient color. The global ambient color is the overall color added to the areas that are not illuminated. Without this color, these areas appear black.

To select the projection mode:

From the **projection mode** pulldown menu, select:

- **render camera** to use the projection type of the render camera. This option is selected by default.
- **perspective** to have the camera's focal length and aperture define the illusion of depth for the objects in front of the camera.
- **orthographic** to use orthographic projection (projection onto the projection plane using parallel rays).
- **uv** to have every object render its UV space into the output format. You can use this option to cook out texture maps.
- **spherical** to have the entire 360-degree world rendered as a spherical map.

To change the global ambient color:

Drag the **ambient** slider, or enter a value between 0 (black) and 1 (white) in the input field.

To render pixels beyond the edges of the frame:

Use the **overscan** slider or field to set the maximum additional pixels to render beyond the left/right and top/bottom of the frame. Rendering pixels beyond the edges of the frame can be useful if subsequent nodes need to have access outside the frame. For example, a Blur node down the node tree may produce better results around the edges of the frame if **overscan** is used. Similarly, a subsequent LensDistortion node may require the use of **overscan**.

16 WORKING WITH STEREOSCOPIC PROJECTS

The title of this chapter is slightly misleading, as Nuke isn't actually limited to stereoscopic views, but rather provides multi-view support for as many views as you need. The views do not have to be stereo pairs, but since that is the most obvious application, this chapter mainly deals with stereoscopic projects.

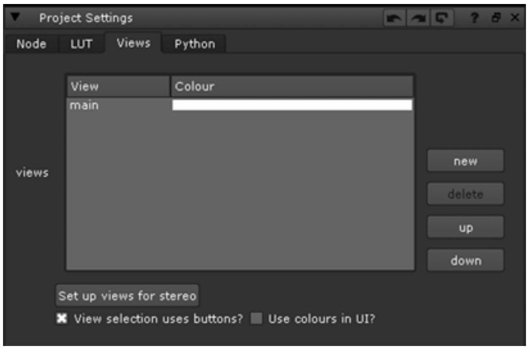
In many ways, Nuke lets you work on stereoscopic material just like you would on any other images. However, there are also a few stereo-specific settings and nodes that you need to be aware of when compositing stereoscopic material. The following teaches you how to set up your stereo project, read in and view your images, use the stereo nodes, and render the final output.

Setting Up Views for the Script

Before you start working on stereoscopic images, you need to set different views for the right and the left eye in the project settings. This allows you to process the individual views separately or both views together, and see the effect of your changes on each view.

To set up views for your project:

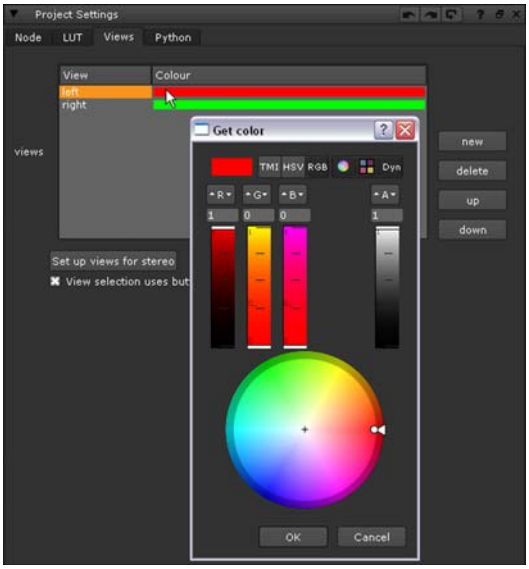
1. Select **Edit** > **Project settings**.
2. Go to the **Views** tab. The available views are listed in the **views** field.



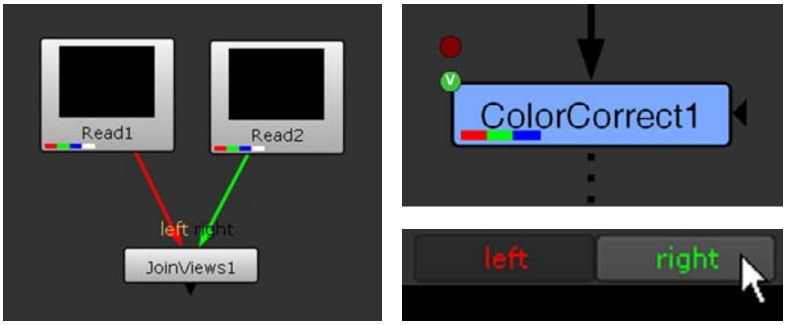
View	Colour
main	

☒ View selection uses buttons? ☐ Use colours in UI?

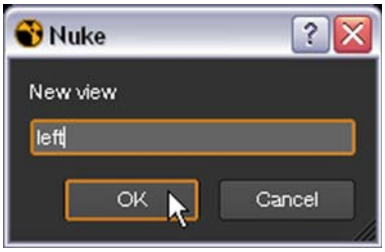
3. If you want to remove the view called **main** and add views called **left** and **right**, click the **Set up views for stereo** button. The two views are assigned colors. To change the colors, double-click on the color field and choose another color from the color picker that opens.


Project Settings
Node
LUT
Views
Python
View
Colour
left
right
views
Set up views for stereo
✖ View selection uses but
Get color
TMI HSV RGB
Dyn
R G B A
1 0 0 1
new
delete
up
down
OK
Cancel

If you check **Use colors in UI?**, these colors are used in Node Graph connections, split views indicators on nodes, and Viewer and ShuffleViews node controls to make it easier to differentiate between views.

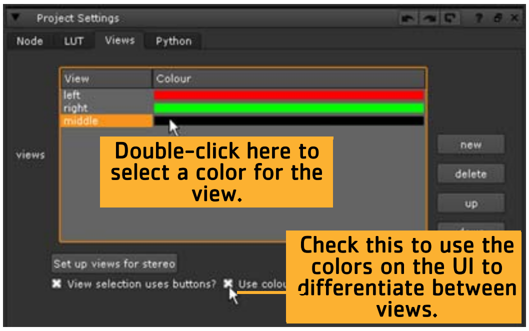

Read1
Read2
left right
JoinViews1
V
ColorCorrect1
left
right

4. If you want to add other new views, click the **new** button.
5. In the dialog that opens, enter a name for the view, for example **middle**. Click **OK**.


Nuke
OK
Cancel

New view: left

6. Repeat steps 4 and 5 as necessary until you've got the views you want. You can assign colors to all views by double-clicking the area on the right of the view name.



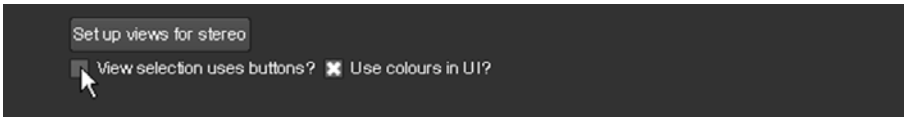
Double-click here to select a color for the view.

Check this to use the colors on the UI to differentiate between views.

7. To delete an unnecessary view, select the view from the list and click the **delete** button. Note that deleting a view does not remove references to it from the script, and any nodes that refer to the deleted view will produce an error.

You can now access the views in your project from the **view** pulldown menu of certain nodes' controls. You'll also notice that each view has its own button in the Viewer controls.

If you created many views, you may want them to appear in a pulldown menu rather than as separate buttons in the Viewer and node controls. To switch to using pulldown menus, uncheck **View selection uses buttons?** on the **Views** tab of the Project settings properties panel.



☐ View selection uses buttons? ☒ Use colours in UI?

Figure 16.1: Setting the way views are selected in the Viewer.

If you are likely to need the same views in several projects, you may want to save the views you created in a template.nk script file. For more information on how to do this, see "Template Scripts" on page 644.

Loading Multi-View Images

Once you have set up the views, you are ready to read your images into Nuke. To make things easier, the images you read in should have the view name or the first letter of the view name in the filename, for example **filename.left.0001.exr**, **filename.l.exr**, or **lefteyefilename.0001.cin**.

If you are using .exr files, your files can contain both the input for the left eye and the input for the right eye, as .exr files support multiple views in a single file. With any other file types, you need to have separate files for the left and right inputs.

To read images in:

1. Select **Image** > **Read.**
2. Navigate to the files containing the images intended for either the left or right eye (or in the case of exr images, both eyes), and select **Open**.
3. Do one of the following:
   - If the images you want to read in contain a view name or the initial letter of one (for example, **left**, **right**, **l** or **r**) in their filenames, replace this with the variable **%V** or **%v** in the **file** field of the Read node's controls. Use **%V** to replace an entire view name (for example, **left** or **right**), and **%v** to replace an initial letter (for example, **l** or **r**). When a variable is used, Nuke reads in the missing inputs and combines all inputs into a single output.

     For example, if you read in **image.left.cin** and changed the name to **image.%V.cin**, Nuke would read in both **image.left.cin** and **image.right.cin** with the same Read node, provided that views called **left** and **right** existed in your project settings. Both input images would be combined into a single output.

Note *If you're using Mac or Linux, you'll need to make sure you use lower case l and r letters in naming your left and right views, as* ***%v*** *variable doesn't recognize upper case L and R letters. On Windows, you can use either.*

     You can also use the **%V** and **%v** variables at a directory level. For example, let's say you have set up views called **testleft**, **testmiddle** and **testright**, and you have the following directories and files:

     mydirectory/testleft/image.testleft.cin

     mydirectory/testmiddle/image.testmiddle.cin

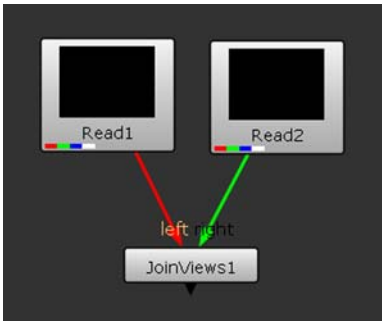
     mydirectory/testright/image.testright.cin.

     If you now read in image.testleft.cin and changed the pathname to **mydirectory/%V/image.%V.cin**, all three inputs would be read in with the same Read node.
   - If the images you want to read in do NOT contain a view name or the initial letter of one (for example, **left**, **right**, **l** or **r**) in the filenames and are not stereo .exr files, insert a Read node for each input and combine them into a single output using the JoinViews node (see below for instructions on how to do that).

- If the images you want to read in are in the stereo .exr file format, you do not need to do anything. However, remember that not all .exr files are stereo .exrs. If you are using ones that are not, follow the instructions in the first two points.

To combine different views into a single output when the views are not indicated in the filenames:

1. Select **Image** > **Read** to read in your image sequences containing the different views.
2. To insert a JoinViews node, select **Views** > **JoinViews**.
3. Connect the inputs of the JoinViews node into the appropriate Read nodes. There should be an input for each view you have created in the project settings. The inputs are labeled with the name of the view.



If you have assigned colors to the views and checked **Use colors in UI?** on the **Views** tab of your project settings, the connecting arrows will reflect the view colors. If this does not happen and the arrows are black, you may have connected the inputs the wrong way around. Check that you have connected each Read node to the correct input of the JoinViews node.

Nuke combines the inputs into a single output.

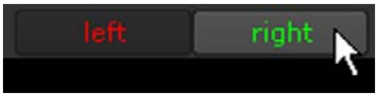
Displaying Views in the Viewer

You can only display the views that exist in your project settings. To see a list of these views or add or delete views, select **Edit** > **Project settings** and go to the **Views** tab. For more information, see "Setting Up Views for the Script" on page 494.

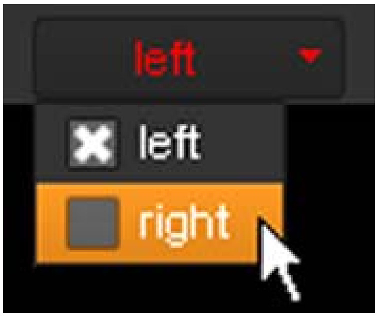
To display a particular view:

1. Add a Viewer into your script if you haven't already done so.
2. On top of the Viewer controls, do one of the following:

- If you have checked **View selection uses buttons?** in the project settings, click the button of the view you want to display. For example, click the **right** button (assuming you have a view called **right** in your script).



- If you haven't checked **View selection uses buttons?** in the project settings, select the view you want to display from the pulldown menu.

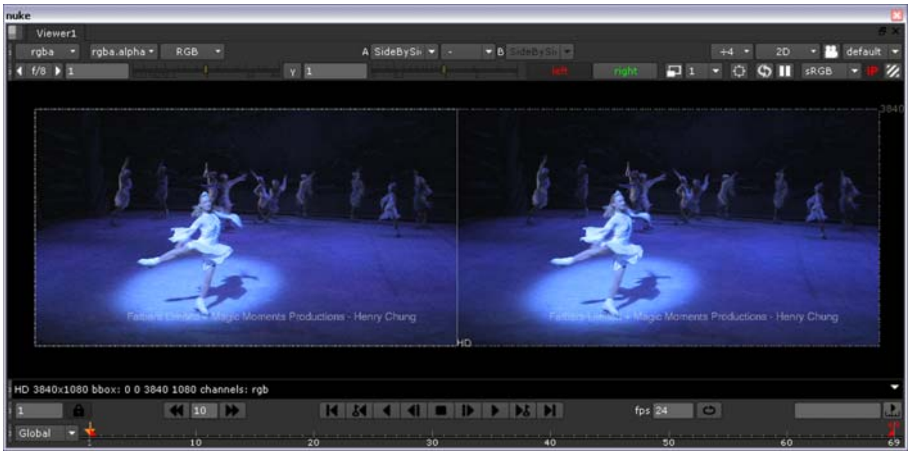


Tip *You can also press the ; (semicolon) and ' (forward single quote) keys to move between different views in the Viewer.*

To display two views next to each other:

1. Add a Viewer into your script if you haven't already done so.
2. If necessary, combine your views into a single output using the JoinViews node. For more information on how to do this, see page 498.
3. Select **Views** > **Stereo** > **SideBySide** to insert a SideBySide node in an appropriate place in your script.
4. In the SideBySide node's controls, select the two views you want to display from the **view1** and **view2** pulldown menus. View1 will be displayed on the left and view2 on the right.
5. If you want to display one view on top of another rather than next to it, check **vertical**. View1 will be displayed above view2.
6. If you want to swap the views around in the Viewer, click the **swap** button.

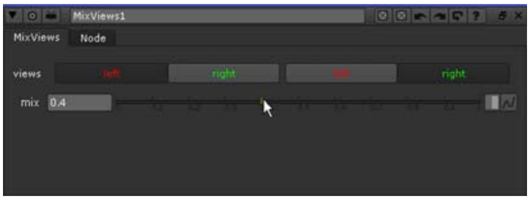
The Viewer displays the two selected views simultaneously, so you can easily compare them.



```
HD 3840x1080 bbox: 0 0 3840 1080 channels: rgb
```

To display a blend between two views:

1. Add a Viewer into your script if you haven't already done so.
2. If necessary, combine your views into a single output using the JoinViews node. For more information on how to do this, see page 498.
3. Select **Views** > **Stereo** > **MixViews** to insert a MixViews node into your script. This node displays a blend between two views in the Viewer, allowing you to check how elements in these views are aligned.
4. In the MixViews controls, use the **views** buttons or pulldown menus to select the two views to blend between.
5. To control the blend between the views, adjust the **mix** slider. Setting the slider to 0 or 1 displays only one of the views. Values between 0 and 1 produce different blends between the views.



views left right left right
mix 0.4

Selecting which Views to Apply Changes To

By default, Nuke applies any changes you make to all views of the processed node. To apply changes to a particular view only (for example, the left view but not the right), you must first do one of the following:

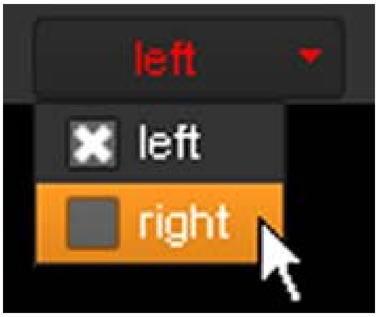
- In the case of most nodes, split the view off in the node's controls.
- In the case of RotoPaint nodes, select the view you want to process from the **view** pulldown menu in the node's controls.

These methods are useful, for example, when you want to perform the same operation on both views but use different values for each.

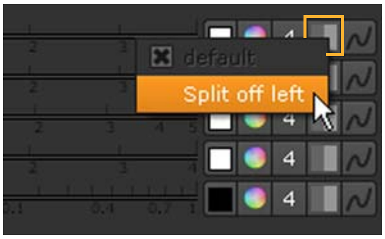
Splitting Views Off

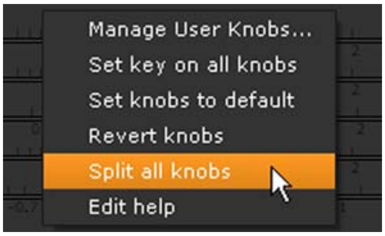
To split a view off:

1. Insert a process node (for example, ColorCorrect) in the appropriate place in your script.
2. If you haven't already done so, attach a Viewer to the node. From the Viewer's controls, select the view you want to make changes to.

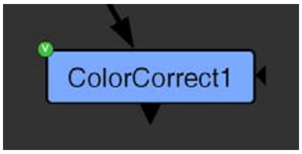

left
☒ left
☐ right

3. Open the node's controls.
4. Click the view button next to the control you want to adjust. From the menu that opens, select **Split off [view name]**. For example, to apply changes to a view called **left**, select **Split off left**. You can also split all the node's controls by selecting **Split all knobs** from the right-click menu.

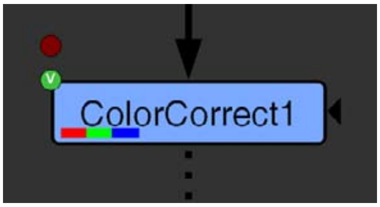

☒ default
Split off left


Manage User Knobs...
Set key on all knobs
Set knobs to default
Revert knobs
Split all knobs
Edit help

An eye appears on the view button and the node gets a small green dot on it in the Node Graph to indicate that views have been split off.



If you have assigned colors to the views and checked **Use colors in UI?** in your project settings, dots also appear on the node to indicate which views have been split off. For example, if you are using red for the left view and split off that view, a red dot appears on the node.



Any changes you now make using the control in question are only applied to the view you chose to split off. Changes to controls that have not been split off are still applied to all views.

To show separate values for each view:

Once you have split off a view, you can apply changes to the existing views separately. Simply click on the small arrow on the left side of a control you have split off. This divides the control so that you can give separate values for each view.

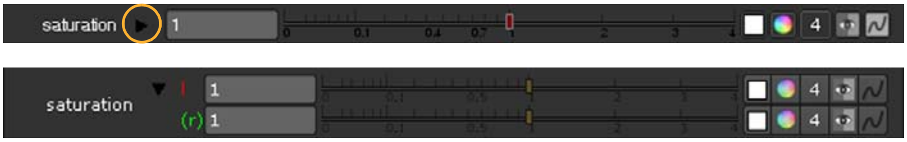


Figure 16.2: Adjusting a split control for only the split view and for all views separately.

To unsplit views:

1. In the node's controls, click the view button.
2. From the menu that opens, select **Unsplit [view]**. For example, to unsplit a view called **left**, you'd select **Unsplit left**.
3. Repeat step 2 for all views you want to unsplit.

The view is unsplit, and all changes you made after splitting it off are lost.

Selecting the View to Process When Using the RotoPaint Node

To select the view to process:

1. Open the RotoPaint node's controls.
2. From the **view** pulldown menu, select the view you want to process. To apply changes to all views at the same time, select all the views separately.
3. If you selected to process just one view, make sure you are viewing the selected view in the Viewer when making your changes.

Performing Different Actions on Different Views

In case you need to perform totally different actions on the two views, you can add a OneView node to separate one view for processing.

To extract a view for processing:

1. Select **Views** > **OneView** to insert a OneView node in an appropriate place in your script.
2. In the OneView node's controls, select the view you want to make changes to from the **view** pulldown menu.

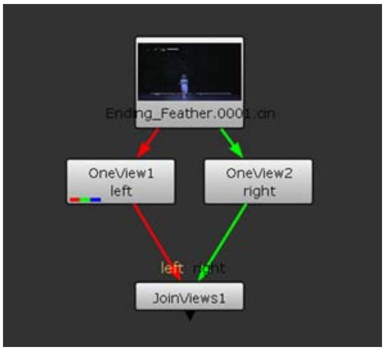
All views are extracted, and any changes you make are only applied to the view you selected (regardless of which view you are displaying in the Viewer).

To make changes to a different view, select it from the OneView node's **view** pulldown menu.

To merge views from two separate streams, select **Views** > **JoinViews** to combine the views (or delete the OneView node from your script).

If you need to extract all views, process them individually, and then merge them together, use the **Split and Join** menu item. This menu item is actually a combination of the OneView and JoinViews nodes. It first extracts all the views you have set up in your project settings and then merges them back together. It's no different to using several OneView nodes together with a JoinViews node, but makes working faster, because you do not need to add each node in a separate go. To use the menu item, select **Views** > **Split and Join**.

For example, if you have created views called **left** and **right** in your project settings and use a **Split and Join** menu item after your Read node, you get the following node tree:



You can then add any necessary nodes, such as color corrections, between the OneView and JoinViews nodes.

Reproducing Changes Made to One View

When rotoscoping, creating paint effects, or doing other operations dependent on image locality, you can have changes made to one view automatically reproduced in the other. This applies to the RotoPaint node, and any nodes, groups, or gizmos that have controls for x and y coordinates.

To reproduce changes made with the above nodes, groups, or gizmos, you need a disparity field that maps the location of a pixel in one view to the location of its corresponding pixel in the other view. You can create a disparity field using The Foundry's O_DisparityGenerator plug-in, which is included in the Ocula plug-in set, or a 3D application. Once you have the disparity field, you can store it in the channels of an .exr file or use the ShuffleCopy node to add the disparity channels in the data stream where you need them.

If you have Ocula installed, you can choose between reproducing your changes using the disparity field or Ocula. If you select to use Ocula, extra refinements are done when correlating the changes from one view to the other. This can improve the results when working with live action footage. When working with CG images, however, the disparity maps should be accurate to begin with and produce good results even without Ocula.

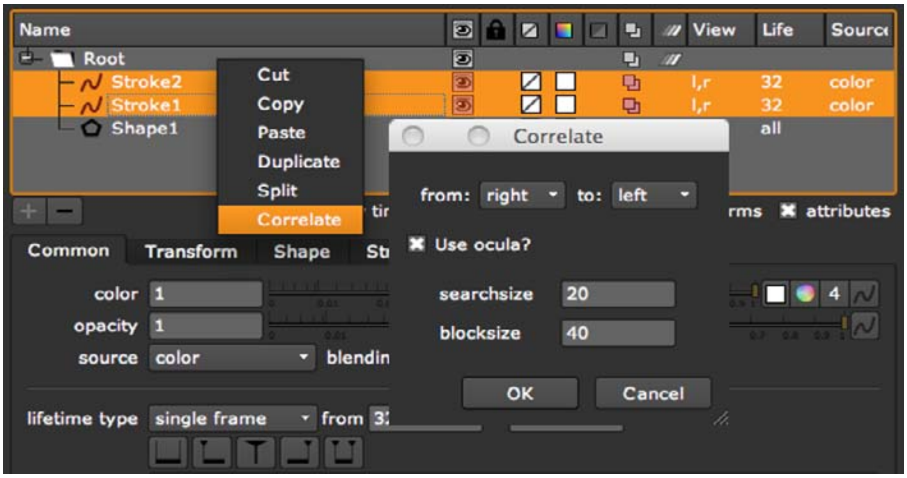
Reproducing Paint Strokes, Beziers, and B-spline Shapes

To create a paint stroke, Bezier or a B-spline shape on one view and have it automatically generated for the other:

1. Make sure there is a disparity field upstream from the image sequence you want to paint on. If the image sequence is an .exr file, the disparity field can be included in its channels. Otherwise, you can use a Shuffle-

Copy node or Ocula's O_DisparityGenerator plug-in to add it in the data stream.

2. To insert a RotoPaint node after the image sequence you want to draw on, click **Draw > RotoPaint**.


Name
View
Life
Source
Root
Stroke2
Stroke1
Shape1
l,r
32
color
all
Cut
Copy
Paste
Duplicate
Split
Correlate
Common
Transform
Shape
color
opacity
source
lifetime type
single frame
attributes
OK
Cancel

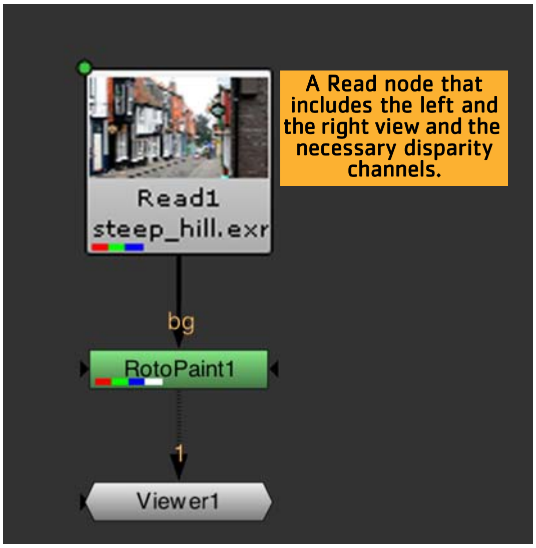
Correlate

from: right to: left

☒ Use ocula?

searchsize 20

blocksize 40


Read1
steep_hill.exr
bg
RotoPaint1
1
Viewer1

A Read node that includes the left and the right view and the necessary disparity channels.

3. In the RotoPaint node controls, select **all** from the **view** pulldown menu. Display the view you want to paint on in the Viewer.
4. With the RotoPaint controls open, draw a stroke/shape in the Viewer.
5. Select the stroke/shape in the stroke/shape list in the RotoPaint node controls.
6. Right-click the stroke/shape and select **Correlate**. The **Correlate** dialog displays.

7. In the Correlate dialog, select which view to correlate from in the **from** dropdown. Similarly, choose which view you want to correlate to in the **to** dropdown. For example, if your stroke/shape was in the correct position in the left view but not the right, you'd select **from: left** and **to: right**.

   This adds the disparity vectors in the map to the current values, creating the corresponding stroke/shape for the other view.

   If you have The Foundry's Ocula plug-ins installed, you can also check the **Use Ocula?** box. This way, extra refinements are made when creating the corresponding stroke/shape, and the results may be more accurate.

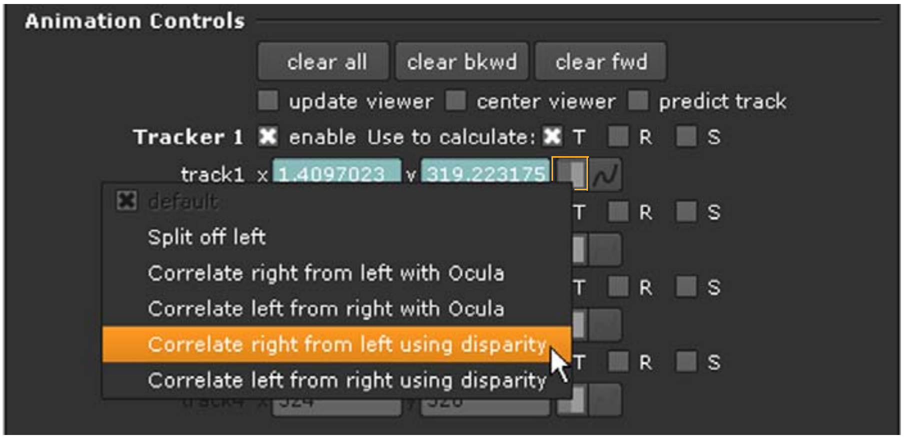
   With Ocula installed, you can also use the **block size** and **search size** controls on the **Correlate** dialog.
8. In the Viewer, switch between the two views to compare the original and the correlated strokes/shapes.
9. If you want to adjust the stroke/shape further, you need to adjust both views independently. Adjustments you make to one view are not automatically generated for the other.

Reproducing X and Y Values

Whenever there are values in any x or y control in Nuke for one view, you can automatically generate the corresponding values for the other view. This is true for both nodes and gizmos. For example, you can use a Tracker node to track something in one view, and then have the track's x and y position generated for the other view automatically.

To produce x and y values for one view and have them automatically generated for the other:

1. Make sure there is a disparity field upstream from the image sequence you are manipulating. If the image sequence is an .exr file, the disparity field can be included in its channels. Otherwise, you can use a ShuffleCopy node or Ocula's O_DisparityGenerator plug-in to add it in the data stream.
2. Insert a node that has an x and y control after the image sequence you are manipulating.
3. Attach a Viewer to the node you added in the previous step, and make your changes in one view.
4. From the View menu next to the **x** and **y** controls, select **Correlate [view] from [view] using disparity**, for example **Correlate right from left using disparity**. This generates the corresponding x and y values for the other view.


Animation Controls
clear all
clear bkwd
clear fwd
update viewer
center viewer
predict track
Tracker 1
enable
Use to calculate:
T
R
S
default
Split off left
Correlate right from left with Ocula
Correlate left from right with Ocula
Correlate right from left using disparity
Correlate left from right using disparity

track1 x 1.4097023 y 319.223175

If you have The Foundry's Ocula plug-ins installed, you can also select **Correlate [view] from [view] with Ocula**. This way, extra refinements are made when creating the corresponding x and y values, and the results may be more accurate.

5. If you want to adjust the x and y values further, you need to adjust both views independently. Adjustments you make to one view are not automatically generated for the other.

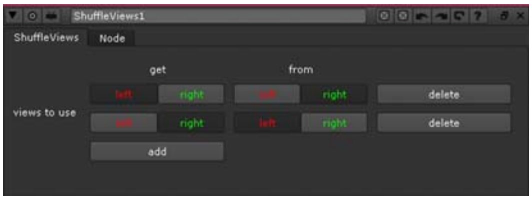
Swapping Views

You can rearrange the views in your script using the ShuffleViews node. For example, you can swap the left and right views around in the pipeline, so that Nuke uses the left input for the right eye and vice versa.

To rearrange views:

1. Select **Views** > **ShuffleViews** to insert a ShuffleViews node in an appropriate place in your script.
2. In the ShuffleViews controls, click **add** as necessary.
3. Use the buttons or pulldown menus to select which view to replace with which. For example, to swap the left and right views around, you need to make the following selections:
   - On one row, select **left** under **get**, and **right** under **from** ("get left from right"). The left view is now replaced with the right view.
   - On another row, select **right** under **get**, and **left** under **from** ("get right from left").

     The right view is replaced with the left view.


ShuffleViews1
ShuffleViews
Node
get
from
views to use
left
right
left
right
delete
left
right
left
right
delete
add

If there aren't enough rows of buttons or pulldown menus on the ShuffleViews node's properties panel, click the **add** button to add a row.

To remove unnecessary rows in the ShuffleViews node's controls, click the **delete** button next to the row you want to remove.

Converting Images into Anaglyph

You can use the Anaglyph node to convert your inputs into anaglyph images, which produce a 3D effect when viewed with 2-color anaglyph glasses.

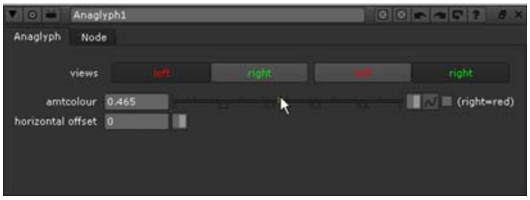
To convert your images into anaglyph:

1. Select **Views** > **Stereo** > **Anaglyph** to insert an Anaglyph node in an appropriate place in your script.
2. Use the **views** controls in the Anaglyph properties panel to select which views you want to use for the left and the right eye.

   Nuke converts the input images into grayscale anaglyph images. The left input is filtered to remove blue and green, and the right view to remove red.

Fatbars Limited + Magic Moments Productions - Henry Chung

3. To add color into the images, drag right on the **amtcolor** slider, or insert a value between 0 (grayscale) and 1 (colored) into the **amtcolor** input field.


Anaglyph1
Anaglyph
Node
views
left
right
left
right
(right=red)

amtcolour 0.465

horizontal offset 0

If the images include areas that are very red, green, or blue, adding more color into them may not produce the best possible results.

Fatbars Limited - Magic Moments Productions - Henry Chung

4. To invert the colors and use the red channel from the right input and the blue and green channels from the left, check the **(right=red)** box.

Fatbars Limited - Magic Moments Productions - Henry Chung

5. To control where the images appear in relation to the screen when viewed with anaglyph glasses, enter a value in the **horizontal offset** input field. To have the images appear in front of the screen, you would usually enter a negative value. To have the images appear further away, you would usually enter a positive value. (This is not the case if you have swapped the left and right views around.)

Tip *If you like, you can register the Anaglyph node as a Viewer Process. This way, you will always have it as a viewing option in the Viewer's Viewer Process menu and can apply it to the current Viewer without having to insert the node in the Node Graph. Do the following:*

1. Create a file called menu.py in your plug-in path directory if one doesn't already exist. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. To register the Anaglyph node as a Viewer Process, save the following in your menu.py:

nuke.ViewerProcess.register("Anaglyph", nuke.createNode, ("Anaglyph", ""))

3. Restart Nuke.

4. To apply the Anaglyph Viewer Process, select it from the Viewer Process menu in the Viewer controls.

*5. To adjust the Anaglyph Viewer Process controls, select **show panel** from the Viewer Process menu.*

For more information on Viewer Processes, see "Input Process and Viewer Process controls" on page 103 and "Creating Custom Viewer Processes" on page 651.

Changing Convergence

The ReConverge node lets you shift *convergence* (the inward rotation of the eyes or cameras) so that any selected point in the image appears at screen depth when viewed with 3D glasses. This point is called the *convergence point.* It is the point where the lines of sight from the two cameras meet.

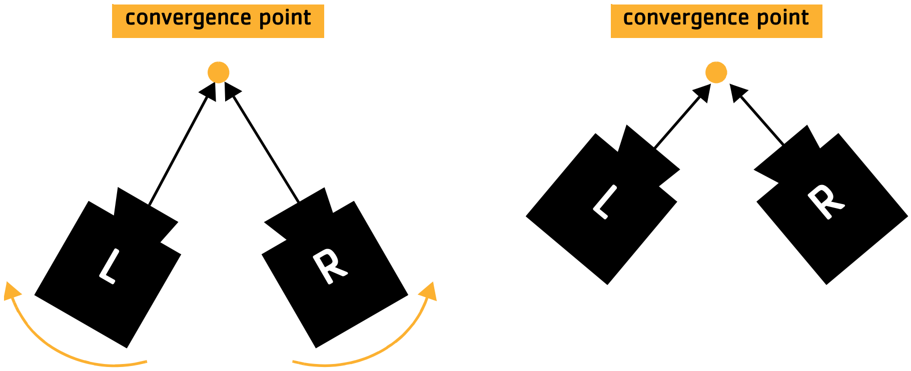


Figure 16.3: Changing convergence moves the point where the lines of sight from the two cameras meet.

At the convergence point, the different views in the image are aligned and appear at screen depth when viewed with 3D glasses. Anything behind the convergence point appears behind the screen, while anything in front of it seems to pop out of the screen. This is illustrated in Figure 16.4.

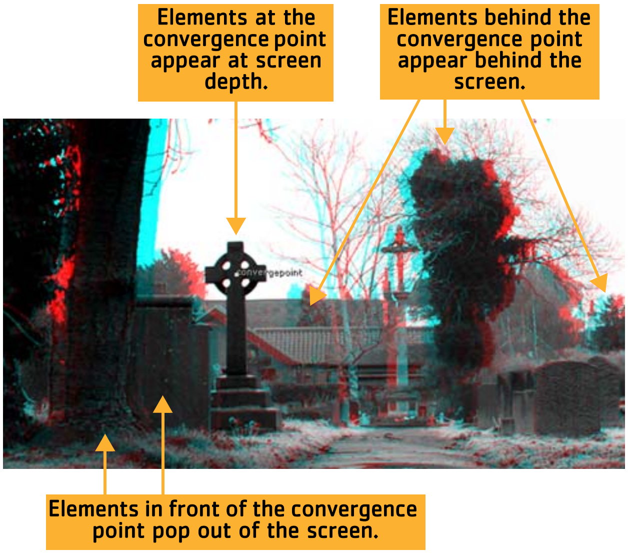

Elements at the convergence point appear at screen depth.
Elements behind the convergence point appear behind the screen.
Elements in front of the convergence point pop out of the screen.

Figure 16.4: Convergence controls where elements in the image appear in relation to the screen when viewed with 3D glasses.

Changing convergence changes the perceived depth of the images. It moves all the elements in the image backwards or forwards a fixed distance while keeping the distance between them the same. This is illustrated in Figure 16.5, where the gray rectangles represent elements depicted in a stereo image.

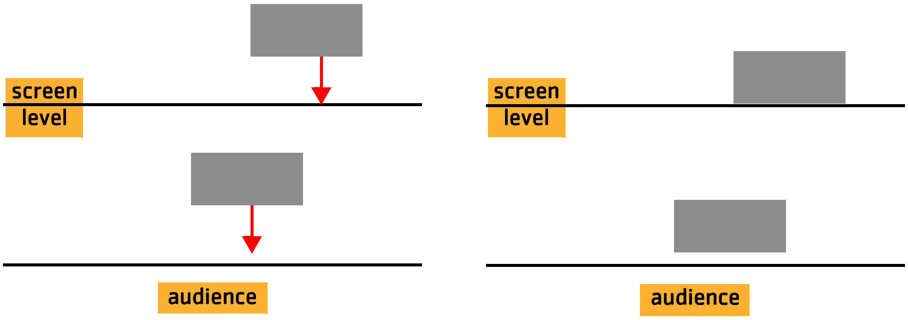

screen level
audience
screen level
audience

Figure 16.5: Changing convergence changes the perceived depth of the images.

Often, the element of an image that appears closest to the audience is used as the convergence point. However, to make an element in your image jump out of the screen, you need to converge on something behind this element.

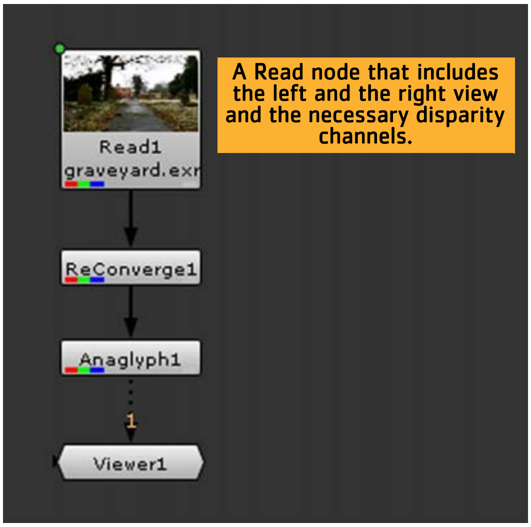
To calculate the convergence shift, the ReConverge node needs a disparity field that maps the location of a pixel in one view to the location of its corresponding pixel in the other view. To create the disparity field, you can use The Foundry's O_DisparityGenerator plug-in, which is part of the Ocula plug-in set. Alternatively, you can create the disparity field in a 3D application. Once you have the disparity field, you can store it in the channels of an .exr file or use the ShuffleCopy node to add the disparity channels in the data stream where you need them.

If you have Ocula installed, you can choose to use it when changing convergence. If you select to use Ocula, extra refinements are done when performing the convergence shift. This can improve the results when working with live action footage. When working with CG images, however, the disparity fields should be accurate to begin with and produce good results even without Ocula.

Note that the ReConverge node only shifts views horizontally, not vertically.

To change the convergence point of a stereo image:

1. Make sure there is a disparity field upstream from the image sequence whose convergence you want to change. If the image sequence is an .exr file, the disparity field can be included in its channels. Otherwise, you can use a ShuffleCopy node or Ocula's O_DisparityGenerator plug-in to add it in the data stream.
2. From the Toolbar, select **Views** > **Stereo** > **ReConverge** to insert a ReConverge node after the image sequence whose convergence you want to adjust.
3. Attach a Viewer to the ReConverge node.
4. To better view the effect of the ReConverge node, insert an Anaglyph node (**Views** > **Stereo** > **Anaglyph**) between the ReConverge node and the Viewer.



A Read node that includes the left and the right view and the necessary disparity channels.

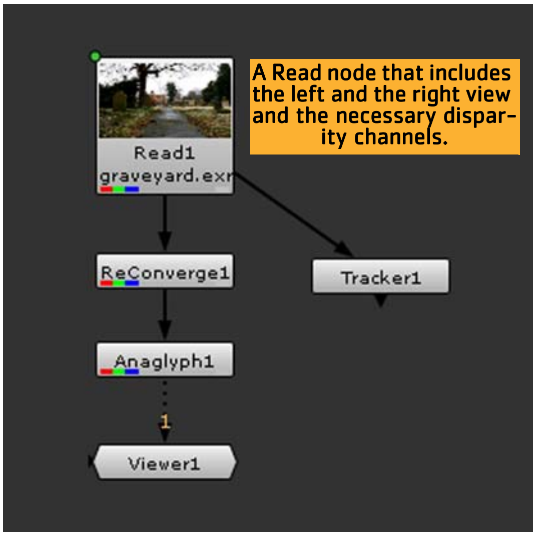
5. If you have The Foundry's Ocula plug-ins installed, check **Use Ocula if available** In the ReConverge properties panel. This way, extra refinements are made when changing the convergence and the results may be more accurate.
6. Make sure the ReConverge properties panel is open. You should see the convergence point overlay in the Viewer. Drag the point on top of the point you want to appear at screen level when viewed with 3D glasses. The convergence shifts to this location.

   You can also move the convergence point by entering the point's x and y coordinates in the **Convergence upon** fields.
7. By default, the ReConverge node moves the right view to achieve the convergence shift. However, if you like, you can use the **Mode** pulldown menu in the ReConverge controls to move the left view instead (select **shift left**) or move both views equally (select **shift both**).
8. If necessary, adjust the offset for convergence (in pixels) in the ReConverge controls. To bring all elements of your image forward from the screen level, enter a positive value in the **Convergence offset** field. To move all elements further away, enter a negative value.

It is also possible to use the same element as the convergence point throughout the image sequence. You can, for example, have the same actor always appear at screen depth. To converge on the same element throughout the sequence, link the ReConverge node with a Tracker node.

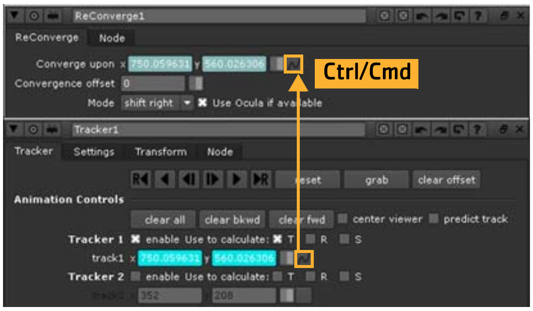
To use the same element as the convergence point throughout the image sequence:

1. Insert a Tracker node after the image sequence whose convergence you want to adjust.


Read1
graveyard.exr
ReConverge1
Tracker1
Anaglyph1
1
Viewer1

A Read node that includes the left and the right view and the necessary disparity channels.

2. Track the point that you want to appear at screen level throughout the sequence. For more information on how to use the Tracker node, refer to Chapter 6: *Tracking and Stabilizing* on page 246.
3. When you have the track animation data, apply it to the O_ReConverge node's **Converge upon** control via a linking expression. The easiest way to do this is to **Ctrl/Cmd**+drag the animation button next to the **track1** controls on top of the animation button next to the **Convergence upon** control.


ReConverge1
ReConverge
Node
Tracker1
Tracker
Settings
Transform
Node
reset
grab
clear offset
Animation Controls
clear all
clear bkwd
clear fwd

Ctrl/Cmd

Converge upon x 750.059631 y 560.026306
Convergence offset 0
Mode shift right ☒ Use Ocula if available

☐ center viewer ☐ predict track
Tracker 1 ☒ enable Use to calculate: ☒ T ☐ R ☐ S
track1 x 750.059631 y 560.026306
Tracker 2 ☐ enable Use to calculate: ☐ T ☐ R ☐ S
track2 x 352 y 208

Previewing and Rendering Stereoscopic Images

On Windows, Linux, and Mac OS X 10.5 (Leopard) or higher, you can preview stereo images using FrameCycler.

Flipbooking Stereo Images within FrameCycler

To flipbook stereo images inside FrameCycler:

1. Select the node whose output you want to preview.
2. Choose **Render** > **Flipbook Selected** from the main menu bar.

   A dialog opens.
3. In the **frame range** field, enter the frame range you want to preview (for example, 1-35 or 1-8 10 12-15).
4. Using the **Views** control, select the two views to flipbook. Click **OK**.

   Nuke renders the selected views as a temporary sequence using the frame range and resolution defined in the script's settings. This may take a few moments.
5. Once the render is complete, Nuke launches FrameCycler and loads in the temporary sequence. You can play it back and view it using FrameCycler's media controls. There are a few different options that allow you to select how to display the stereo images. To learn more about them, refer to the documentation provided with FrameCycler.

Rendering Stereoscopic Images

You can render several views using a single Write node. When using the stereo extensions for the .exr file format, Nuke writes the output of both views into a single file. With any other file types, the views are written into their respective files.

To render .exr files:

1. Select **Image** > **Write** to insert a Write node in an appropriate place in your script.
2. In the Write node's controls, select **exr** from the **file type** pulldown menu.
3. From the **views** pulldown menu, select the view(s) you want to render, for example **left, right**.
4. Adjust any other Write controls as necessary and click **Render**. Nuke prompts you for the frames to render.

Nuke writes several views into a single file.

To render files that are not in the .exr file format:

1. Select **Image** > **Write** to insert a Write node in an appropriate place in your script.
2. In the Write nodes' controls, select the file type of your images from the **file type** pulldown menu.
3. When entering names for the rendered image sequences, you can use the variable **%V** (with a capital V) to represent the words **left** and **right** (or any other full view names) in the filenames, for example **filename.%V.####.exr**. To represent the letters **l** and **r** (or the first letters of any views), use the variable **%v** (with a lower-case v) instead. When rendering, Nuke then fills this in with left, right, l, or r, and renders all views you specify in the next step.
4. Adjust any other Write controls as necessary and click **Render**. Nuke prompts you for the frames to render as well as the views to execute (assuming you have set up several views in the project settings).

Nuke renders several views, but writes them into separate files. If you did not specify a view in the filenames (using either the name of the view, its first letter, or a variable), you can only render one view.

Note *For command line renders, you can pass the -**view** argument with a list of view names to render, separated by a comma. If you do not specify a -**view** argument, Nuke renders all views.*

17 PREVIEWS AND RENDERING

Nuke supports a fast, high-quality internal renderer, with superior color resolution and dynamic range without a slowdown in the workflow. These are some of the key features of Nuke's rendering engine:

- Multi-threaded rendering to take advantage of multiple processors in its calculations.
- Scanline (as opposed to buffer-based) rendering allows you to immediately see portions of render output.
- Calculations performed with 32-bit precision, using linear light levels.

This chapter teaches you how to use the renderer's various features to preview a script's output and generate its final elements. You'll also learn how to preview output on an external broadcast video monitor.

Previewing Output

This section explains how to preview individual frames in a Nuke Viewer window, how to render a flipbook for a sequence of frames, and how to preview output on an external broadcast video monitor.

Previewing in a Nuke Viewer

When you connect a Viewer to a given node's output (by selecting the node and pressing a number key), Nuke immediately starts rendering the output in the Viewer using all available local processors.

Keep in mind the following tips in order to speed up this type of preview rendering:

- First, if you don't need to evaluate the whole image, zoom into the area of interest (see "Zooming" on page 59 for more information). Nuke will then render only the portion of scan lines visible within the Viewer.
- Alternatively, you can use the Viewer's region of interest (ROI) feature to render only a portion of the image, while seeing that result in the context of the whole image.

To enable the ROI render feature:

1. Press **Alt+W** over the Viewer. The Viewer's **ROI** button turns red, indicating that the feature is enabled.
2. Drag on the Viewer to draw the region of interest. The Viewer will now render only the pixels within the region.

To edit the position or size of current ROI:

1. Click the **ROI** button so that it turns red. The overlay for the current ROI appears in the Viewer.
2. To reposition the ROI:

   Using the crosshair in the middle of the ROI, drag the ROI to the desired location.
3. To resize the ROI:

   Drag any corner or side of the ROI until you achieve the desired size.

To disable the ROI render feature:

Click the Viewer's **ROI** button. It turns gray, signalling that it is off. The Viewer will now render all of the visible image.

Flipbooking Sequences

Flipbooking a sequence refers to rendering out range of images (typically at proxy resolution), then playing them back in order to accurately access the motion characteristics of added effects.

You have a few options for flipbooking within Nuke:

- You can enable automatic disk caching of rendered frames, then play these frames back using Nuke's native Viewer. This option does *not* let you define a specific playback rate.
- You can render out a temporary image sequences to FrameCycler, a RAM-buffering playback utility which is automatically installed with your copy of Nuke and plays back sequences at the defined frame rate.
- You can also set up an external flipbooking application in Nuke using Python. For more information on how to do this, see "Flipbooking with External Applications" on page 572.

Flipbooking within Nuke

The Nuke Viewer automatically saves to disk a version of every frame it displays. When you play through sequences in the Viewer, it reads, where possible, from this cache of prerendered images, making real-time play back possible (depending, of course, on image resolution and your hardware configuration). You can define the location and size of the Viewer cache in the Preferences.

Depending on what **gl buffer depth** has been set to in the Viewer settings, the cache can contain 8-bit (**byte**), 16-bit (**half-float**), or 32-bit (**float**) image data. This offers a trade-off between speed and quality. Half-float and float modes provide higher precision than byte but are also slower to process.

To set the location and size of the Viewer cache:

1. Click **Edit > Preferences** to display the Preferences dialog.
2. In the **disk cache** field, enter the pathname of the directory in which you want to store the flipbook images (for example, c:/temp).
3. Using the **disk cache size** control, select the number of gigabytes you want to allow the image cache to consume.
4. Click the **Save Prefs** button to update preferences and then restart Nuke.

The Viewer will now cache each frame it displays in the directory specified. When you click the playback buttons on the Viewer, or drag on the scrub bar, Nuke will read in images from this cache.

Note that the cached images have unique names reflecting their point of output location in the script. This means that you can cache images from multiple nodes in the script without overwriting previously cached images.

For more information on caching, see "Image Caching" on page 129.

Flipbooking within FrameCycler

To flipbook an image sequence inside FrameCycler, do the following.

1. Select the node whose output you wish to see flipbooked.

Note *If you select a Write node in the step above, you must first click its **Render** button in order to manually render its output to the destination defined in the **file** field. This step is necessary only in the case of Write nodes.*

2. Select **Render** > **Flipbook selected** (or press **Alt**+**F**).
   Alternatively you can click the **Flipbook this Viewer** button in the Viewer you want to flipbook.

   A dialog opens.
3. Check that your settings are correct in the dialog. The default values are copied from the Viewer you currently have active. You can change them if necessary:
   - **Flipbook** - set the flipbooking application you want to use.
   - **Take settings from** - set which Viewer should be used to draw default values from.
   - **Enable ROI** - Check to define your region of interest. For more information, see "Region of interest (ROI)" on page 101.

- **Channels -** select which channel set to display in the flipbook result.
- **Frame range -** set the frame range you want to flipbook.
- **Use proxy -** check to use proxy mode.
- **Render in background** - check to render in the background. If you check this, you can also set **#CPU limit** and **Memory limit** controls. The former limits the number of threads that Nuke will use in the background and the latter limits the amout of cache memory that nuke will use.

Note *If you're rendering multiple sequences in the background, this can take up more than the total RAM on your machine. When running background renders of any type, you need to make sure they don't require more RAM all together than what's available on the machine, otherwise you may experience problems such as hanging.*

- **Continue on error** - check to keep rendering even if an error occurs during the process.
- **Delete existing temporary files -** Check to delete any existing temporary files with the same file name before flipbooking.
- **LUT** - select the LUT appropriate for viewing. By default, the flipbook will render your files with a linear colorspace and attempt to pass a LUT file to the flipbook.
- **Burn in the LUT** - If you check this box the flipbook files will be rendered with the LUT applied. If you don't check it, the flipbook will display using it's equivalent LUT (based on the LUT's name). If you have an equivalent LUT available in the flipbook program, then it's better not to check the Burn in the LUT box. This way, when you measure pixel values in the flipbook application they match what you get in the Nuke Viewer.
- **Views** - set which stereo views to include.

4. Click **OK**.

   Nuke renders as a temporary sequence the output of the selected node using the frame range and resolution defined in the script's settings. This may take a few moments.

5. Once the render is complete, Nuke launches FrameCycler and loads in the temporary sequence. You can play it back and view it using FrameCycler's media controls.

Note *FrameCycler comes packed with many features to complement flipbooking. You can, for example, attach sound files to the image sequence, cut it, or splice it together with other image sequences. Surf to **www.iridas.com** for more information.*

Previewing on an External Broadcast Video Monitor

To check the final result in correct video colorspace and pixel aspect ratio, you can preview the current Viewer image on an external broadcast video monitor. This option is only available in 32- and 64-bit Windows and 32-bit Mac OS X versions of Nuke, and requires additional hardware, such as a monitor output card or a FireWire port. Our monitor output architecture is designed to support any monitor output card that works as a QuickTime Video Output Component. For obvious reasons of practicality we are unable to test every possible configuration. Currently, development and testing is done using BlackMagic Decklink HD Extreme 2 and AJA Kona/Xena LHe.

Nuke can output images to external broadcast monitors in either 8- or 10-bit RGB color modes. 10-bit color is automatically selected when the Viewer's **gl buffer depth** setting is **half-float** or **float**, provided it is supported by the monitor output card. In all other cases, 8-bit color is used. Please note that selecting a monitor output mode with the phrase 10 bit in its description will only output true 10-bit color if a **gl buffer depth** of either **half-float** or **float** is selected. Half-float and float modes are considerably slower to process, so it is recommended to stick with **byte** mode for monitor output unless 10-bit color is specifically required.

To preview output on an external broadcast video monitor:

1. Press S on the Viewer to open the Viewer settings.
2. From the **monitor output device** menu, select the external device you want to use. All available devices are automatically detected and listed in this menu.
3. From the **monitor output mode** menu, select the display mode for the device you selected in the previous step. The available options depend on the device you are using. By default, the most recently used display mode is chosen.
4. Press Ctrl/Cmd+U, or click on the monitor output button in the Viewer controls.

   Alternatively, you can check **enable monitor output** in the Viewer settings.

From now on, any output connected to the Viewer will be sent to the monitor output device you selected.

The monitor output image is always full frame, unzoomed (1:1 ratio), and unpanned, regardless of what the Viewer it is linked to is set to. This means that slightly mismatching formats (for example, 640x512 / 640x448 for PAL/NTSC) will not be rescaled to fit the monitor.

If you save your Nuke script with monitor output enabled, this setting is saved with the script. The next time you open the same script, monitor output will be enabled.

To disable viewing on an external broadcast video monitor, do one of the following:

- Click on the monitor output button in the Viewer controls.
- Select **Viewer** > **Toggle Monitor Output** from the menu bar.
- Press Ctrl/Cmd+U. This can be particularly useful if you are using a monitor output device that can draw things on the screen you are using to display the Nuke window (such as the Digital Cinema Desktop Preview device installed with Apple Final Cut Pro).
- Press S on the Viewer to open the Viewer settings, and uncheck **enable monitor output**.

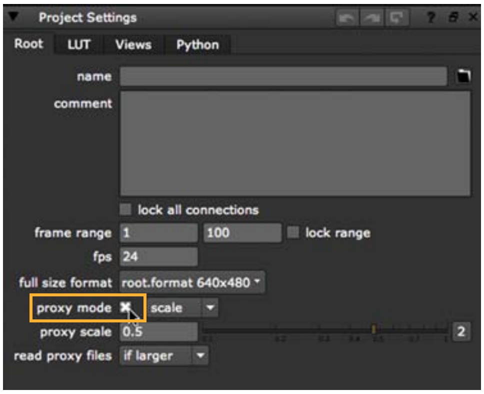
Rendering Output

Nuke can render images locally— on your workstation— or it can be setup to render images on a network render farm. Before rendering, verify that your project settings have the correct output format and proxy format selected.

Render Resolution and Format

Before rendering a script, it's important to check what is the currently active mode: the full-size or the proxy mode. Nuke executes all renders at the currently active scale. Thus, when a script is rendered in proxy mode, processing will be done at the proxy scale and image output will go to the file name in the Write node's **proxy** field. If you do not specify a proxy file name, the render will fail with an error. It never resizes the proxy image, and it will not write the proxy image over the full-size one.

To view and change the proxy resolution for the current script file, choose **Edit > Project settings** from the menu bar, or press **S** with the mouse pointer over the Node Graph or the Properties Bin.



name
comment
☐ lock all connections
frame range 1 100 ☐ lock range
fps 24
full size format root.format 640x480
proxy mode ☒ scale
proxy scale 0.5 2
read proxy files if larger

Figure 17.1: Changing the output resolution under Project settings.

From the Project settings properties panel, you can select a new render format from the list of predefined resolutions, and toggle proxy rendering. You can also choose the **new** option under either **full size format** or **proxy format** or use the **proxy scale** fields to define custom render resolutions for the composite. When rendering in proxy mode, use the pulldown menu on the right to select whether to use the resolution defined under **proxy format** or **proxy scale**. Also check that you have set **read proxy file** to what you want - this setting controls how Read nodes choose the file to read (full res file or proxy) in the proxy mode. For more information on these settings, refer to “Proxy format and proxy scale” on page 124 and “Read nodes and proxy files” on page 126.

Output (Write) Nodes

With the correct resolution and format selected, you then insert Write nodes to indicate where you want to render images from the script.

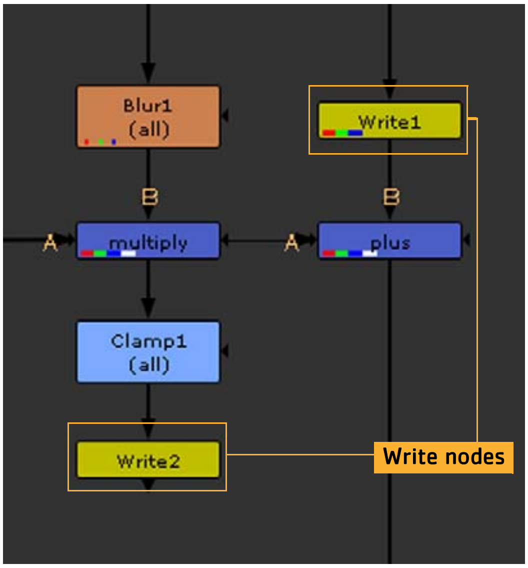


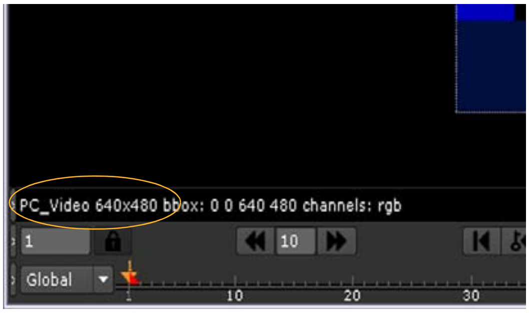
Figure 17.2: Inserting Write nodes for rendering.

One Write node is usually placed at the bottom of the compositing tree to render the final output. However, Write nodes have both input and output connectors, so they may be embedded anywhere in the compositing tree.

You can execute renders for a single Write node or all Write nodes in your compositing script.

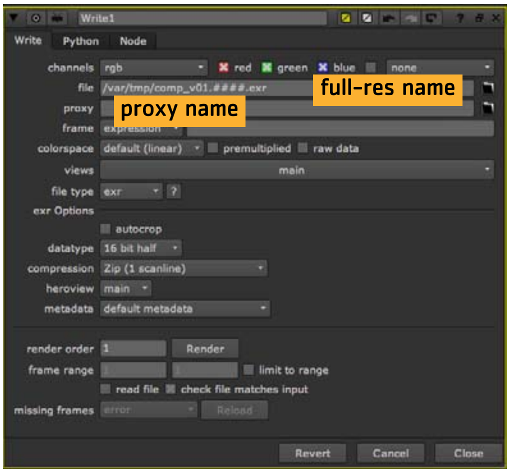
To render a single Write node:

1. Select the node in the script from which you want to render an image.
2. Choose **Image > Write** (or press **W** over the Node Graph). Nuke attaches a Write node and opens its properties panel.
3. Connect a Viewer to the Write node you want to render and verify that the correct resolution is displayed for output. If necessary, press **Ctrl/Cmd+P** to toggle between full-res and proxy resolution. The displayed output resolution will be used for rendering.



PC_Video 640x480 bbox: 0 0 640 480 channels: rgb

4. In the properties panel, click the **file** or **proxy** field's folder icon (depending on whether you want to render high res or low res images) and browse to the directory where you want to store the rendered sequence. For instructions on using the file browser, see "Using the File Browser" on page 111.



Write1

channels: rgb ☒ red ☒ green ☒ blue ☐ none
file: /var/tmp/comp_v01.####.exr
proxy:
frame: expression
colorspace: default (linear) ☐ premultiplied ☐ raw data
views: main
file type: exr
exr Options
☐ autocrop
datatype: 16 bit half
compression: Zip (1 scanline)
heroview: main
metadata: default metadata
render order: 1
frame range: 1 1 ☐ limit to range
☐ read file ☐ check file matches input
missing frames: error

5. After the path, type a name for the rendered image and then click **OK**. If you're rendering an image sequence, include the frame number variable (for example, **####**) in the name.

   See "To render selected or all Write nodes in the script:" below for examples of valid file names with the frame number variable.

6. If necessary, adjust the following controls:

   - Using the **channels** pulldown menu and checkboxes, select the channels you want to render.
   - Using the **frame** pulldown menu and input field, set the relation between the currently processed frame and the numbering of the frame written out. For more information, see "Changing the Numbering of Rendered Frames" on page 529.

- check **read file** if you want the output of the Write node to be produced by reading the rendered file back in rather than by processing the upstream node tree. For more information on this and the **missing frames** control, see "Using a Write Node to Read in the Rendered Image" on page 531.
- From the **colorspace** pulldown menu, select which lookup table to use when converting between the images' color space and Nuke's internal color space.
- From the **file type** pulldown menu, select the file format for the rendered images. If you don't specify a file format, Nuke uses the extension in the file name to figure out the format.
- Check the **limit to range** box if you want to disable the node when executed outside of a specified frame range. In the **frame range** fields, enter the range of frames you want to make executable with this Write node.

7. In the Write node properties, click the **Render** button.
8. Nuke prompts for a frame range, defaulting to the range you gave in the **frame range** fields. If necessary, change the start and end frames (for example, **1-100**), and then click **OK.**

Tip *When specifying the frame range to render, you can enter complex frame ranges into the frame range prompt dialog. For example if you enter "1-5 8 10 15 22-25", it will only render those frames. Likewise, you can specify multiple ranges on the command line, for example:*

```
nuke -F 1-5 -F 8 -F 10 -F 15 -F 22-25 -x myscript.nk
```

You can see the progress of your render in the status window that appears. When the render is complete, the rendered images are added to the directory you specified in step 4.

Tip *When rendering with the Write node, you can force a certain data type by adding the data type and a colon before the file path. For example, you can enter ftiff:C:\Temp\test.tif as the file path to render a file whose data type is ftiff and extension tif.*

Tip *If you are rendering .mov files, you can choose the QuickTime codec from the **codec** pulldown menu, and adjust advanced codec options by clicking the **advanced** button.*

Note *QuickTime is only supported by default on 32-bit Windows and Mac OS X.*

On 64-bit Windows, you cannot load QuickTime files. As a workaround, you can use the 32-bit version of Nuke to load your files and convert them to another file format.

Note *When writing QuickTime files, some users have reported receiving the following error at the end of rendering:*

Failed to flatten movie data: the movie is open in another application.

*This is because the output file has been opened by some other process. This can be the Finder on Mac OS X trying to show a preview, an OS file system search trying to index the file, or a virus checker, for example. The workaround is to turn off **Fast Start** in the Write node controls to skip the flattening process that is affected.*

To render selected or all Write nodes in the script:

1. Connect a Viewer to a Write node you want to render and verify that the correct resolution is displayed for output.
2. If necessary, press **Ctrl/Cmd+P** to toggle between full-res and proxy resolution. The displayed output resolution will be used for rendering.
3. If you want, you can change the order in which your Write nodes are rendered by giving them custom render order numbers in the **render order** field.
4. Do one of the following:
   - With the desired Write node selected, choose **Render** > **Render selected** (or press **F7**).
   - Choose **Render** > **Render all** (or press **F5**).
5. In the Render dialog, adjust the render settings if necessary. The default values are drawn from the Viewer you have active.
   - **Frame range** - set the frame range you want to render.
   - **Use proxy** - check to use proxy mode.
   - **Render in background** - check to render in the background. If you check this, you can also set **#CPU limit** and **Memory limit** controls. The former limits the number of threads that Nuke will use in the background and the latter limits the amout of cache memory that nuke will use.

     Setting the number of CPUs is useful for limiting Nuke to use only a particular number of CPUs (for example, 4 threads can occupy 4 CPUs) with the OS trying to distribute time accordingly, or you could limit Nuke to a number of CPUs so background rendering won't interfere too much with your interactive Nuke.
   - **Continue on error** - check to keep rendering even if an error occurs during the process.

- **Views** - set which stereo views to include in the render.

6. Click **OK**.

Tip *When specifying the frame range to render, you can enter complex frame ranges into the frame range prompt dialog. For example if you enter "1-5 8 10 15 22-25", it will only render those frames. Likewise, you can specify multiple ranges on the command line, for example:*

```
nuke -F 1-5 -F 8 -F 10 -F 15 -F 22-25 -x myscript.nk
```

File Name Conventions for Rendered Images

There is no parameter in the Write node to specify output format. Instead, format is indicated by a prefix or an extension when you type the file name. Here is the appropriate syntax:

```
<prefix>:/<path>/<name>.<frame number variable>.<extension>
```

The optional *<prefix>:* can be any valid extension. *<path>* is the full pathname to the directory where you want to render. The *<frame number variable>* is usually entered as **####**, with #### indicating frame numbers padded to four digits.

You can change the padding by substituting #### with any number of hash marks. For example, two-digit padding would be **##**, three-digit would be **###**, and five-digit would be **#####**.

With these conventions in mind, suppose you want to save an image sequence called “final_comp_v01” to the TIFF16 format. Here are examples of names that will work in the Write node:

```
tiff16:/<path>/final_comp_v01.####.tiff
/<path>/final_comp_v01.####.tiff16
tiff16:/<path>/final_comp_v01.####
```

All extensions supported by Nuke may be used for the prefix. See “Appendix B: Supported File Formats” on page 829 for a complete list of recognized extensions.

When a prefix is used, it takes precedence over the format represented by an extension. In fact, the prefix makes the extension unnecessary.

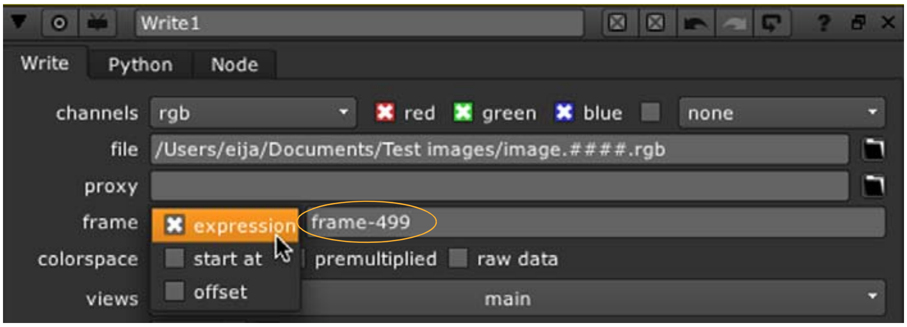
You could, for example, enter **exr:/<path>/###** as the file name, and this would create an OpenEXR image sequence with frame numbers only, padded to three digits.

Changing the Numbering of Rendered Frames

By default, when you render an image sequence, Nuke assumes an exact relation between the currently processed frame and the numbering of the frame written out. Therefore, if you choose to render frame 15, the resulting file is numbered accordingly, for example image.0015.rgb. However, the **frame** parameter on the Write node lets you change this behavior. For instance, if you have a sequence that runs from image.0500.rgb to image.1000.rgb, you may want to render it so that the frame numbering in the resulting files runs from image.0001.rgb to image.0501.rgb. You can do so via expressions, specified start frames, and constant offsets. Each method is described below.

Using expressions

1. Select **Image** > **Write** to insert a Write node into your script.
2. In the Write properties panel, click the **file** folder icon, then navigate to the directory path where you want to save the rendered image sequence. Enter a name for the image sequence.
3. Set **frame** to **expression**. Enter an expression in the field on the right. The expression changes the relation between the currently processed frame and the numbering of the frame written out. The resulting file name for the current frame is displayed on the Write node in the Node Graph.

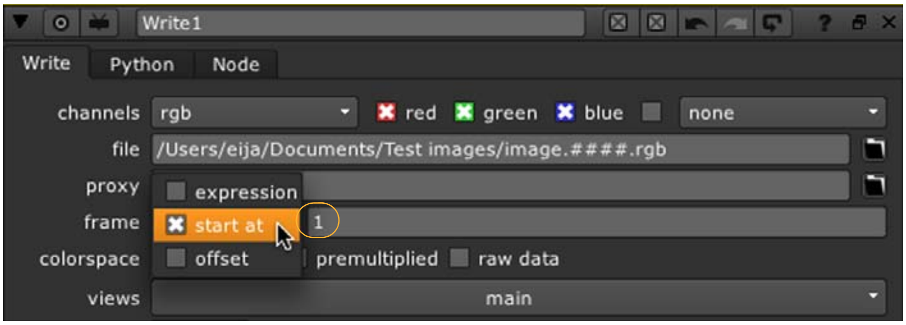

Write1
channels rgb ☒ red ☒ green ☒ blue ☐ none
file /Users/eija/Documents/Test images/image.####.rgb
proxy
frame ☒ expression frame-499

For example, if your clip begins from frame 500 and you want to name that frame image.0001.rgb rather than image.0500.rgb, you can use the expression **frame-499**. This way, 499 frames are subtracted from the current frame to get the number for the frame written out. Frame 500 is written out as image.0001.rgb, frame 501 is written out as image.0002.rgb, and so on.

Another example of an expression is **frame*2**. This expression multiplies the current frame by two to get the number of the frame that's written out. At frame 1, image.0002.rgb is written out; at frame 2, image.0004.rgb is written out; at frame 3, image.0006.rgb is written out; and so on.

Specifying a frame number for the first frame in the clip

1. Select **Image** > **Write** to insert a Write node into your script.
2. In the Write properties panel, click the **file** folder icon, then navigate to the directory path where you want to save the rendered image sequence. Enter a name for the image sequence.
3. Select **start at** from the **frame** menu. Enter a start frame number in the field on the right. This specifies the frame number given to the first frame in the sequence. The numbering of the rest of the frames is offset accordingly.

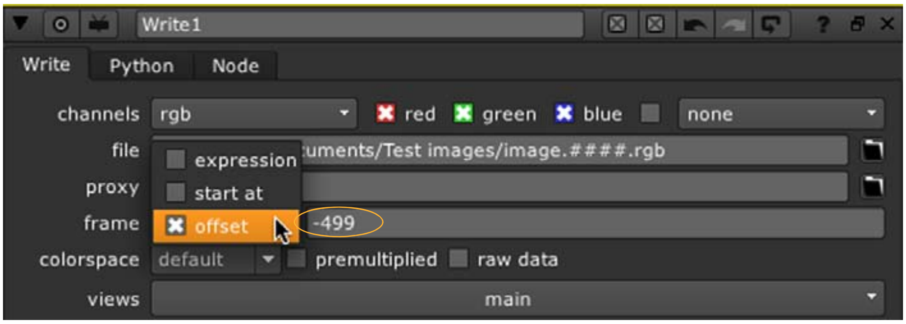


channels rgb ☒ red ☒ green ☒ blue ☐ none
file /Users/eija/Documents/Test images/image.####.rgb
proxy
frame ☒ start at 1
☐ expression
☐ offset
colorspace
☐ premultiplied ☐ raw data
views main

For example, if your sequence begins from frame 500 and you enter **1** in the field, frame 500 is written out as image.0001.rgb, frame 501 as image.0002.rgb, and so on. Similarly, if you enter **100** in the field, frame 500 is written out as image.0100.rgb.

Offsetting all frame numbers by a constant value

1. Select **Image** > **Write** to insert a Write node into your script.
2. In the Write properties panel, click the **file** folder icon, then navigate to the directory path where you want to save the rendered image sequence. Enter a name for the image sequence.
3. From the **frame** menu, select **offset**. Enter a constant offset in the field on the right. This constant value is added to the current frame to get the number for the frame that's written out.



channels: rgb ☒ red ☒ green ☒ blue ☐ none
file: …uments/Test images/image.####.rgb
proxy:
frame: ☒ offset -499
colorspace: default ☐ premultiplied ☐ raw data
views: main

For example, if your clip begins from frame 500 and you want to render this first frame as image.0001.rgb rather than image.0500.rgb, you can use **-499** as the constant offset. This way, 499 is subtracted from the current frame to get the number for the frame that's written out. At frame 500, image.0001.rgb is written out; at frame 501, image.0002 is written out, and so on.

Using a Write Node to Read in the Rendered Image

You can use a Write node to both render an image and read the rendered image back in. Because reading the output of a Write node from a file is faster than calculating its output by processing the node tree upstream, this can be particularly useful on large comps. When you have finished working on a branch of the node tree, you can insert a Write node after it, render the output, and use the same Write node to read the rendered image in. If you later need to edit the nodes upstream, simply make your changes and render the Write node again to update the image being read in.

To use a Write node to read in the rendered image:

1. Render an image as described in "To render a single Write node:" on page 524. We recommend rendering the image as an exr. This way, Nuke writes the hash value of the incoming node tree into the rendered file. If the node tree changes and the rendered file gets out of date, the hashes won't match and Nuke will notify you of the problem.

What is the hash value?

The hash value is a unique number (for example, b1c9c0aff2012a8) calculated from a node and the entire tree of nodes connected to its input. The class of the node and all the current control settings contribute to the hash value.

You can display the hash value at any point in the node tree by selecting a node in the Node Graph and pressing I. The hash will be different at different points in the tree.

2. In the Write node properties, check **read file**. When this is on, Nuke ignores the upstream node tree and uses the rendered image as the output of the Write node.
3. To check whether the input file is up to date with the input tree connected to the Write node, check **check file matches input**. This only works with EXR files written by Nuke and when the proxy mode and down-rez are disabled. If the input file cannot be checked, Nuke will display the word **unchecked** on the Write node in the Node Graph.
4. If there is an error when loading the rendered file, select what to do from the **missing frames** menu:
   - **error** - display an error message on any missing frames.
   - **black** - replace any missing frames with black.
   - **checkerboard** - replace any missing frames with a checkerboard image.
   - **read input** - display the result of the input tree rather than the rendered file on any missing frames.

Tip *You can also use the Precomp node (**Other** > **Precomp**) to reduce portions of the node tree to pre-rendered image inputs. For more information, see "Using the Precomp Node" on page 147.*

Render Farms

Nuke is supported by virtually all third-party and proprietary render-queuing software. By integrating Nuke with such a system, the render load can be distributed across all the Nuke-licensed machines on your network, whether Windows, Mac, or Linux-based.

Your installation of Nuke may be configured to send jobs to a network render farm, which is usually made available under the Render menu (i.e., **Render > Render**). However, because this option must be customized for each studio, you should check with your system administrator for instructions on how to send a Nuke script for network rendering.

18 EXPRESSIONS

This chapter is intended as a primer on how to apply expressions (programmatic commands) to Nuke parameters. It explains how to perform some common tasks with expressions (for example, how to link the values of one parameter to another), and concludes with a table all the functions that you may include as part of an expression.

Understanding Expression Syntax

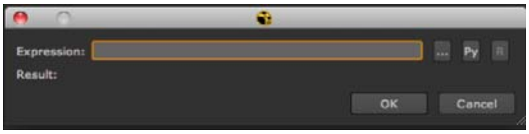
You enter Nuke expressions in the expression dialog, which you can open by pressing the equals sign (=) on a parameter. In the dialog, you can enter text that either references values from other parameters (creating a linking expression) and/or applies mathematical functions of some kind to the current values.

Linking Expressions

Through expressions, you can link the parameters from one node and control the values of the parameters in other nodes. When creating a linking expression, type the elements listed in the table below; remember to separate each element with a period.

Element	Description
Node name	The node with the source parameter (i.e., **Transform1**).
Parameter name	The name of the parameter with the source value (for example, **translate**).The name is defined internally, and may not match the parameter's label that appear in the Nuke interface. If necessary, hover over the parameter's field with your mouse pointer and its name will appear in the pop-up tool tip.
Child parameter name (optional)	Some parameters include child parameters, such as the fields for x and y axes, or red, green, and blue color channels). Child parameter names do match the label that appears before the parameter's field (for example, **x**).
Time (optional)	By default, linking expressions pull values from the current frame number, but you can read values from other frames, either statically or dynamically (that is, with a temporal offset). If you want to read in a static value for a given frame, you just type that frame number inside a set of parenthesis (for example, **(10)**). If you want to read in dynamic values but with an offset in time, type **t**, the variable for time, followed by a **+** (for a forward offset) or **–** (for a backward offset), followed by a number representing the number of frames worth of offset. For example, typing **(t-2)** would capture values that are two frames back from the current frame.

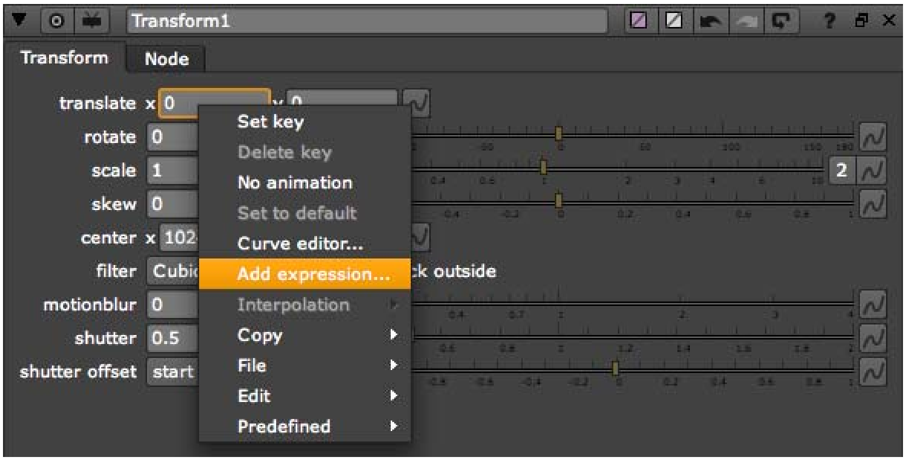
Thus, to create a linking expression that pulls the value from a Transform node's x translation field at the tenth frame, you would type = on a parameter to open the expression dialog, and then enter **Transform1.translate.x(10)** in the dialog's **Expression** field.


Expression:
...
Py
Result:
OK
Cancel

The steps below recap the process for creating a linking expression.

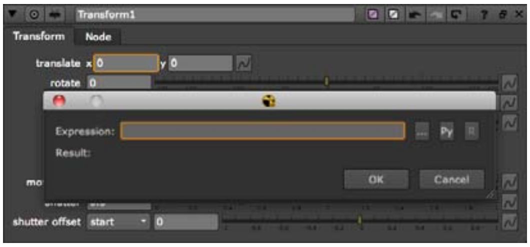
To reference values from another parameter (method #1):

1. Click on the destination parameter (the one which will receive values from another parameter).
2. To display the expression dialog, right-click on the parameter and select **Add expression**,


Transform1
Transform
Node
Set key
Delete key
No animation
Set to default
Curve editor...
Add expression...
Interpolation
Copy
File
Edit
Predefined
k outside

translate x 0
rotate 0
scale 1
skew 0
center x 102
filter Cubi
motionblur 0
shutter 0.5
shutter offset start

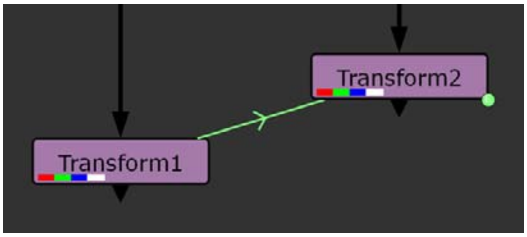
OR type = in the parameter field.


Transform1
Transform
Node
translate x 0 y 0
rotate 0
Expression:
Py
Result:
OK
Cancel
shutter offset start 0

3. In the dialog that opens, type the name of the node containing the source parameter and a period. (Each node prominently displays its name on its face.)
4. If you want to enter a multi-line expression, you can click the multi-line edit field button.
5. Follow the name of the node by the source parameter's name and a period. (If you don't know the parameter's name, you can hover over its field in order to see it displayed in a tool tip.)
6. Optionally, type the child parameter's name and a period.
7. Optionally, type a frame number or offset variable in brackets (for example, **(2)** or **(t-2)**) in order to specify the frame or range of frame from which you pull values.
8. Next to the expression entry field, you can click the **Py** button to automatically make your expression a Python callback. You can also toggle the **R** button to have your expression interpreted as an expression or as a series of statements. For example, with the multi-line edit mode and the Python mode on, you could enter the following expression, and get 15 as the resulting value:

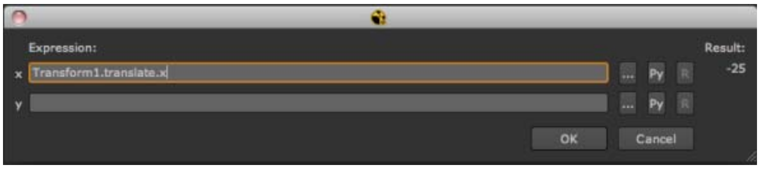
```
-execlocal

def example():
  a = 5
  return a

def example2():
  b = 10
  return b

ret = example()+example2()
```

9. Click **OK**. This links the parameters, which turn blue. In the Node Graph, a green arrow appears between the nodes to indicate that they are linked via an expression.



10. To edit the expression later on, right-click on the parameter and select **Edit expression** (or press = on the parameter). You can

also click the animation button and select **Edit expression** to edit expressions for all the parameters next to the button.



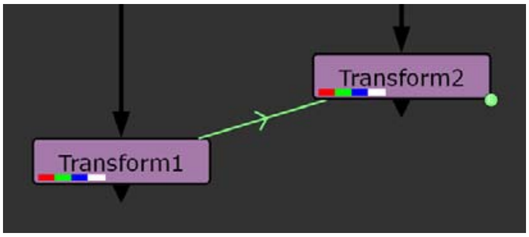
Expression:

x Transform1.translate.x

Result: -25

To reference values from another parameter (method #2):

1. **Ctrl/Cmd**+drag the parameter that has the values you want to use on top of the parameter that will receive these values. This links the parameters, which turn blue. In the Node Graph, a green arrow appears between the nodes to indicate that they are linked via an expression.



   To view or edit the expression, right-click on the parameter and select **Edit expression**.

2. If you want to link several parameters at the same time, **Ctrl/ Cmd**+drag the animation button next to the source parameters on top of the animation button next to the destination parameters.

   To view or edit the expressions used to link the parameters, click the animation button and select **Edit expression**.

To Convert Expressions Between Scripting Languages

Depending on where you need to use an expression, you might find that you want to, for example, use Nuke expressions to TCL expressions or embed Python functions in a Nuke expression. The different languages are used in different parts of Nuke:

- Python can be used in the Script Editor, in the Script Command (**File > Script Command**) and in scripts run when Nuke starts (such as init.py and menu.py). For more information, see "The Script Editor and Python" on page 561.
- TCL can be used in most string knobs (where text other than just numbers can be entered), in the Script Command dialog (**File > Script**

Command), to open some compatibility start up scripts (such as init.tcl and formats.tcl).
- Nuke expressions can be used on the Add Expression dialog with most knobs in Nuke and expression entry field in the Expression node.

You can use the following functions to use different types of expressions together:

- **nuke.expression()** to use a Nuke expression in Python code.
- **expression** to use a Nuke expression in TCL.
- **nuke.tcl()** to run TCL code in Python.
- **python** to run Python code in TCL
- **[ ]** (square brackets) to embed TCL in a Nuke expression (or a string knob)
- **[python {...}]** to embed Python in a Nuke expression.

Tip *Note that putting braces ({ }) around Python code when embedding it in TCL may make the process a bit easier, because this prevents TCL from performing its own evaluation of the Python code before passing it through to the Python interpreter. For example: [python {"hello " + "world"}]*

Tip *Note that the "python" TCL command by default evaluates a single line of code and returns the result. Use the "-exec" option (e.g. "python -exec") if you want to run multiple lines. Please refer to the Nuke TCL Scripting documentation (**Help** > **Documentation** > **TCL Scripting**) for further information.*

Adding Mathematical Functions to Expressions

You can incorporate mathematical functions into parameters. For example, you might negate an expression in order to invert a tracking curve which you wish to use to stabilize an element (such an expression might resemble the following: **-(Transform1.translate.x)**).

You can also rely on a function to add more complex mathematical operation to your expressions. The table below list all the functions which you may incorporate into Nuke expressions.

Function	Purpose	Operator Usage	Related Functions
abs (x)	Returns the absolute value of the floating-point number x.	x	See also: fabs.
acos (x)	Calculates the arc cosine of x; that is the value whose cosine is x.	If x is less than -1 or greater 1, acos returns nan (not a number).	See also: cos, cosh, asin, atan.

Function	Purpose	Operator Usage	Related Functions
asin (x)	Calculates the arc sine of x; that is the value whose sine is x.	If x is less than -1 or greater 1, asin returns nan (not a number)>	See also: sin, sinh, acos, atan.
atan (x)	Calculates the arc tangent of x; that is the value whose tangent is x. The return value will be between -PI/2 and PI/2.	x	See also: tan, tanh, acos, asin, atan2.
atan2 (x, y)	Calculates the arc tangent of the two variables x and y. This function is useful to calculate the angle between to vectors.	x, y	See also: sin, cos, tan, asin, acos, atan, hypot.
ceil (x)	Round x up to the nearest integer.	x	See also: floor, trunc, rint.
clamp (x, min, max)	Return x clamped to [min ... max]	x, min, max	See also: min, max.
clamp (x)	Return x clamped to [0.0 ... 1.0]	x	See also: min, max.
cos (x)	Returns the cosine of x	x in radians	See also: acos, sin, tan, cosh.
cosh (x)	Returns the hyperbolic cosine of x, which is defined mathematically as (exp(x) + exp(-x)) / 2.	x	See also: cos, acos, sinh, tanh.
curve (frame)	Returns the y value of the animation curve at the given frame	optional: frame, defaults to current frame	See also: value, y.
degrees (x)	Convert the angle x from radians into degrees	x	See also: radians.
exp (x)	Returns the value of e (the base of natural logarithms) raised to the power of x.	x	See also: log, log10.
exponent (x)	Exponent of x.	x	See also: mantissa, ldexp.

Function	Purpose	Operator Usage	Related Functions
fBm (x, y, z, octaves, lucanarity, gain)	Fractional Brownian Motion. This is the sum of octaves calls to noise(). For each of them the input point is multiplied by pow(lacunarity,i) and the result is multiplied by pow(gain,i). For normal use, lacunarity should be greater than 1 and gain should be less than 1.	x, y, z, octaves, lacunarity, gain	See also: noise, random, turbulence.
fabs (x)	Returns the absolute value of the floating-point number x.	x	See also: abs.
false ()	Always returns 0		See also: true.
floor (x)	Round x down to the nearest integer.	x	See also: ceil, trunc, rint.
fmod (x, y)	Computes the remainder of dividing x by y. The return value is x - n y, where n is the quotient of x / y, rounded towards zero to an integer.	x, y	See also: ceil, floor.
frame ()	Return the current frame number.		See also: x.
from_byte (color component)	Converts an sRGB pixel value to a linear value.	color_component	See also: to_sRGB, to_rec709f, from_rec709f.
from_rec709f (color component)	Converts a rec709 byte value to a linear brightness	color_component	See also: form_sRGB, to_rec709f.
from_sRGB (color component)	Converts an sRGB pixel value to a linear value.	color_component	See also: to_sRGB, to_rec709f, from_rec709f.
hypot (x, y)	Returns the sqrt(x*x + y*y). This is the length of the hypotenuse of a right-angle triangle with sides of length x and y.	x, y	See also: atan2.
int (x)	Round x to the nearest integer not larger in absolute value.	x	See also: ceil, floor, trunc, rint.

Function	Purpose	Operator Usage	Related Functions
ldexp (x)	Returns the result of multiplying the floating-point number x by 2 raised to the power exp.	x, exp	See also: exponent.
lerp (a, b, x)	Returns a point on the line f(x) where f(0)==a and f(1)==b. Matches the lerp function in other shading languages.	a, b, x	See also: step, smoothstep.
log (x)	Returns the natural logarithm of x.	x	See also: log10, exp.
log10 (x)	Returns the base-10 logarithm of x.	x	See also: log, exp.
logb (x)	same as exponent()	x	See also: mantissa, exponent.
mantissa (x)	Returns the normalized fraction. If the argument x is not zero, the normalized fraction is x times a power of two, and is always in the range 1/2 (inclusive) to 1 (exclusive). If x is zero, then the normalized fraction is zero and exponent() Returns zero.	x	See also: exponent
max (x, y, ...)	return the greatest of all values	x, y, (...)	See also: min, clamp.
min (x, y, ...)	return the smallest of all values	x, y, (...)	See also: max, clamp
mix (a, b, x)	same as lerp()	a, b, x	See also: step, smoothstep, lerp
noise (x, y, z)	creates a 3D Perlin noise value. This produces a signed range centerd on zero. The absolute maximum range is from -1.0 to 1.0. This produces zero at all integers, so you should rotate the coordinates somewhat (add a fraction of y and z to x, etc.) if you want to use this for random number generation.	x, optional y, optional z	See also: random, fBm, turbulence
pi ()	Returns the value for pi (3.141592654...)		
pow (x, y)	Returns the value of x raised to the power of y.	x, y	See also: log, exp, pow

Function	Purpose	Operator Usage	Related Functions
pow2 (x)	Returns the value of x raised to the power of 2.	x, y	See also: pow
radians (x)	convert the angle x from degrees into radians	x	See also: degrees
random (x, y, z)	creates a pseudo random value between 0 and 1. It will always generate the same value for the same x, y and z. Calling random with no arguments will create a different value on every invocation.	optional x, optional y, optional z	See also: noise, fBm, turbulence
rint (x)	Round x to the nearest integer.	x	See also: ceil, floor, int, trunc
sin (x)	Returns the sine of x	x in radians	See also: asin, cos, tan, sinh
sinh (x)	Returns the hyperbolic sine of x, which is defined mathematically as (exp(x) - exp(-x)) / 2.	x	See also: sin, asin, cosh, tanh
smoothstep (a, b, x)	Returns 0 if x is less than a, returns 1 if x is greater or equal to b, returns a smooth cubic interpolation otherwise. Matches the smoothstep function in other shading languages.	a, b, x	See also: step, lerp
sqrt (x)	Returns the non-negative square root of x.	x	See also: pow, pow2
step (a, x)	Returns 0 if x is less than a, returns 1 otherwise. Matches the step function other shading languages.	a, x	See also: smoothstep, lerp
tan (x)	Returns the tangent of x	x in radians	See also: atan, cos, sin, tanh, atan2
tanh (x)	Returns the hyperbolic tangent of x, which is defined mathematically as sinh(x) / cosh(x).	x	See also: tan, atan, sinh, cosh
to_byte (color component)	Converts a floating point pixel value to an 8-bit value that represents that number in sRGB space.	color_component	See also: form_sRGB, to_rec709f, from_rec709f

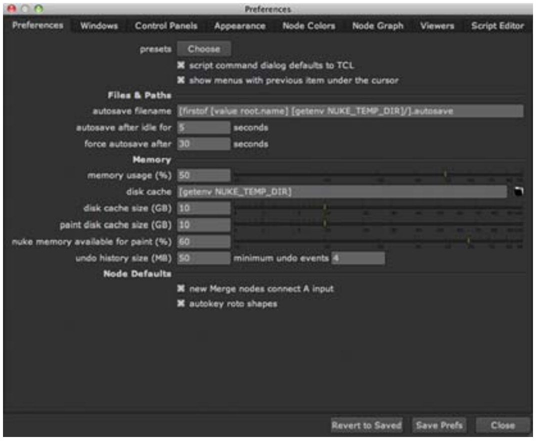
Function	Purpose	Operator Usage	Related Functions
to_rec709f (color component)	Converts a floating point pixel value to an 8-bit value that represents that brightness in the rec709 standard when that standard is mapped to the 0-255 range.	color_component	See also: form_sRGB, from_rec709f
to_sRGB (color component)	Converts a floating point pixel value to an 8-bit value that represents that number in sRGB space.	color_component	See also: form_sRGB, to_rec709f, from_rec709f
true ()	Always Returns 1	See also: false	
trunc (x)	Round x to the nearest integer not larger in absolute value.	x	See also: ceil, floor, int, rint
turbulence (x, y, z, octaves, lucanarity, gain)	This is the same as fBm() except the absolute value of the noise() function is used.	x, y, z, octaves, lucanarity, gain	See also: fBm, noise, random
value (frame)	Evaluates the y value for an animation at the given frame.	optional: frame, defaults to current frame	See also: y, curve
x ()	Return the current frame number.		See also: frame
y (frame)	Evaluates the y value for an animation at the given frame	optional: frame, defaults to current frame	See also: value, curve

19 SETTING INTERFACE PREFERENCES

Nuke offers a highly customizable interface. This chapter teaches you how you can use the preferences dialog to define the color, font, and spatial arrangement of all main interface elements.

Displaying the Preferences Dialog

Choose **Edit** > **Preferences**, or press **Shift+S**. The preferences dialog appears.



presets
☒ script command dialog defaults to TCL
☒ show menus with previous item under the cursor
Files & Paths
autosave filename [firstof [value root.name] [getenv NUKE_TEMP_DIR]/].autosave
autosave after idle for 5 seconds
force autosave after 30 seconds
Memory
memory usage (%) 50
disk cache [getenv NUKE_TEMP_DIR]
disk cache size (GB) 10
paint disk cache size (GB) 10
nuke memory available for paint (%) 60
undo history size (MB) 50 minimum undo events 4
Node Defaults
☒ new Merge nodes connect A input
☒ autokey roto shapes

Figure 19.1: Nuke preferences.

Changing Preferences

The function of each preference is described below under “The Available Preference Settings” on page 544.

When you make a change to a preference, in most cases, the interface registers that change immediately (for example, an interface element displays in the new color).

Saving Preferences

Nuke stores your preference settings in a file called **preferences.nk**, which resides in your Home directory. Each Nuke user can maintain his or her own unique settings.

If you make changes inside the preferences dialog, you need to explicitly save your changes to this file by clicking **Save Prefs**. If you simply close the Preferences dialog, they will only be in effect for your current session. To reset any changes you made and use the preferences saved in preferences.nk, click **Revert to Saved**.

To save your preferences:

Make the desired changes inside the preferences dialog, then click the **Save Prefs** button. Nuke writes the new settings to **preferences6.nk** file, which you can find in the .nuke directory:

- **On Windows**: The .nuke directory can be found under the directory pointed to by the HOME environment variable. If this variable is not set (which is common), the .nuke directory will be under the folder specified by the USERPROFILE environment variable - which is generally of the form *drive letter*:\Documents and Settings\login name\ (Windows XP) or *drive letter*:\Users\login name\ (Windows Vista).

  To find out if the HOME and USERPROFILE environment variables are set and where they are pointing at, enter %HOME% or %USERPROFILE% into the address bar in Windows Explorer. If the environment variable is set, the folder it's pointing at is opened. If it's not set, you will get an error.
- **On Mac OS X**: /Users/*login name*/.nuke
- **On Linux**: /users/*login name*/.nuke

Your new preferences will remain in effect for the current and all subsequent sessions.

Resetting Preferences

To delete your preferences and reset them to default, delete the **preferences6.nk** file. The next time you launch Nuke, it will rebuild the file with the default preferences.

The Available Preference Settings

The preferences dialog is divided into the following tabs:

- **Preferences** - Settings for color schemes, automatic backup of files, memory usage, certain node defaults, and script command dialog defaults.
- **Windows** - Settings for window positions, tool tips, and window snapping.
- **Control Panels** - Settings for changing the behavior of control panels.
- **Appearance** - Settings for changing the colors and font on the application interface.

- **Node Colors** - Settings for changing the colors of different nodes and Viewer overlays.
- **Node Graph** - Settings for changing the appearance of the Node Graph (for example, colors, font, background grid usage, arrow size, and Dot node behavior).
- **Viewers** - Settings for changing the colors, controls, interaction speed, and buffer bit depth of Viewers.
- **Script Editor** - Settings for changing the behavior and syntax highlighting colors of the Script Editor.

The below tables describe the available preference settings and their functions. Each tab of preferences is described in its own table.

Preferences Tab

Setting	Function
presets: Choose	Select a predefined color scheme: **Standard** or **Silver**. You can also create new color schemes yourself. Do the following: 1. Go to the **Appearance** tab and adjust the colors until you are happy with them. Click **Save Prefs**. 2. Make a copy of the preferences6.nk file and rename it to UI_*name*.tcl (where *name* is the name of your color scheme). You can find preferences6.nk in the following location: •On Windows: In the .nuke directory, which can be found under the directory pointed to by the HOME environment variable. If this variable is not set (which is common), the .nuke directory will be under the folder specified by the USERPROFILE environment variable - which is generally of the form "*drive letter*:\Documents and Settings\login name\" (Windows XP) or "*drive letter*:\Users\login name\" (Windows Vista). To find out if the HOME and USERPROFILE environment variables are set and where they are pointing at, enter %HOME% or %USERPROFILE% into the address bar in Windows Explorer. If the environment variable is set, the folder it's pointing at is opened. If it's not set, you get an error. •On Mac OS X: /Users/*login name*/.nuke •On Linux: /users/*login name*/.nuke 3. Put this file into the plug-in path. For more information on the plug-in path, see "Environment Variables" on page 611. When you next launch Nuke, your new color scheme is listed as one of the predefined color schemes.

Setting	Function
Script command dialog defaults to TCL	Select which scripting language the script command dialog defaults to when you select **File** > **Script Command**. When this is checked, the dialog defaults to TCL. When this is NOT checked, it defaults to Python.
show menu with previous item under the cursor	When this is checked, right-click menus are opened with the previously selected item under the cursor. When this is NOT checked, right-click menus are opened with nothing under the cursor.
Files & Paths	
autosave filename	Define where and under what name Nuke saves your automatic backup files. By default, the files are saved with the extension *.autosave* in the same folder as your project files. To change this, enter a full directory pathname in the autosave filename field. You can use *[value root.name]* to refer to the full script pathname, and *[file tail [value root name]]* to just refer to the filename with its extension.
autosave after idle for	Define how long (in seconds) Nuke waits before performing an automatic backup after you have left the system idle (that is, haven't used the mouse or the keyboard). If you set the value to 0, automatic backups are disabled.
force autosave after	Define how long (in seconds) Nuke waits before performing an automatic backup regardless of whether the system is idle. If you set the value to 0, forced automatic backups are disabled.
Memory	
memory usage (%)	Set the amount of RAM that Nuke uses for processing images. Generally, the default setting of 50% gives a good trade off between performance and interactivity.
disk cache	Nuke to saves all recent images displayed in the Viewer for fast playback. Using this control, you can specify where you want Nuke to save these images. Pick a local disk (for example, c:/temp), preferably with the fastest access time available. It's also important to leave enough space for the maximum disk cache size (defined below). The environment variable NUKE_DISK_CACHE can be used to override this setting.
disk cache size (GB)	Select the maximum size the disk cache can reach. Ensure there is enough space on the disk for this to be reached. The environment variable NUKE_DISK_CACHE_GB can be used to override this setting.
paint disk cache size (GB)	Select the maximum size the RotoPaint disk cache can reach. Ensure there is enough space on the disk for this to be reached.

Setting	Function
nuke memory available for paint (%)	Limit the memory usage of the RotoPaint node to a percentage of available Nuke memory.
undo history size (MB)	Select the amount of RAM to use for the undo history. If this limit is exceeded, older items will be discarded.
minimum undo events	Select the minimum number of undo events you want Nuke always to store, even if they breach the memory limit.
Node Defaults	
new Merge nodes connect A input	When this is checked, any Merge nodes you add are connected to the currently selected node via the A input of the Merge node. When this is NOT checked, the B input of the Merge node is used instead.
autokey on Bezier	When this is checked, the **autokey** control in the RotoPaint and Bezier node properties is enabled by default, and Bezier and B-spline shapes and paint strokes are automatically saved as key shapes. When this is NOT checked, the **autokey** control is disabled by default, and Bezier and B-spline shapes and paint strokes are not automatically saved as key shapes.

Windows Tab

Setting	Function
tooltips on	When this is checked, you can hover the cursor over a control to display its tool tip. When this is NOT checked, no tool tips are displayed.
Positions	
stored: Clear	When you rearrange floating windows, such as the color picker window, Nuke remembers their position the next time you open them. You can use this control to clear the remembered positions.

Setting	Function
floating windows	This preference only has an effect on Linux and can be used to fix problems with floating windows. What you should set it to depends on the window manager you are using (GNOME or KDE): • **Utility (GNOME)** - Choose this if you are using the GNOME window manager. If you haven't specifically selected to use KDE, you're most likely using GNOME. If you find that floating windows go behind full screen windows, choose Utility (GNOME). • **Normal (KDE)** - Choose this if you are using the KDE window manager. If this option is not used on KDE, all the floating windows, such as Viewers and Color Pickers, vanish when you move focus away from Nuke. If floating windows (control panels, viewers, curve editor, etc) are disappearing when another application gets focus (using KDE on Linux) set **Hide utility windows for inactive applications** to off (**KDE Menu** > **Settings** > **Desktop** > **Window Behavior** > **Advanced**).
show dialogs under the cursor	When this is checked, pop-up dialogs appear in the current position of the cursor. When this is NOT checked, pop-up dialogs appear in the middle of the Nuke application window.
Snapping	
snap when moving windows	When this is checked and you move floating windows, the windows snap to screen edges and other floating windows, making it easy to place them right next to each other. When this is NOT checked and you move floating windows, the windows do not snap to anything. On Linux, window snapping may not work. However, most Linux window managers let you do window snapping if you hold down **Shift** while moving the window.
snap if parallel without touching	When this is checked, all floating windows' edges are considered to extend infinitely outward, so that you can easily align the windows even if they aren't close to each other. For example, say you have one floating window on the left and another on the right. When you move either of these up or down, their top and bottom edges snap to align with the top or bottom edge of the other window. When this is NOT checked, window snapping is restricted to nearby windows.
threshold	Define how close to each other (in pixels) the windows have to be for them to snap together.
Script Loading	
re-open viewers when loading saved scripts	When this is checked and you open a saved script, any Viewers in the script are opened in the Viewer pane. When this is NOT checked, you need to open the Viewers manually by double-clicking on them in the Node Graph.

Setting	Function
use window layout from saved scripts	When this is checked, Nuke opens saved scripts with the window layout they were saved with. When this is NOT checked, Nuke opens saved scripts with the default layout.
File Browser	
start file browser from most recently user directory	When this is checked, the File Browser will open in the directory that you have used most recently. When this is NOT ticked, the File Browser will open in your default root directory.

Control Panels Tab

Setting	Function
new panels go to	Select where you want control panels to appear when you add a new node or double-click on an existing node in the Node Graph: • **own window** - Each new control panel appears in its own floating window. • **top of control panel bin** - Each new control panel appears on top of the Properties Bin. • **bottom of control panel bin** - Each new control panel appears in the bottom of the Properties Bin.
max nodes in Properties bin	Define the maximum number of control panels that can appear in the Properties Bin at the same time. When you're working, you can override this setting using the field on top of the Properties Bin. However, the number saved in the preferences is used as the default whenever you launch Nuke.
reopen acts like new panel	When this is checked and you reopen a floating control panel, Nuke opens the panel in the same place as a new panel, even if you moved the panel to a new location earlier. When this is NOT checked and you reopen a floating control panel, Nuke remembers where the panel was located when it was last closed and opens it in that position.
double-click moves panel	When this is checked, you can double-click on a node whose control panel is already open to move the control panel to the place where the new panels go. When this is NOT checked, double-clicking on a node whose control panel is already open does change the panel's position but selects the panel and (in the case of floating panels) moves in on top of other panels.

Setting	Function
close Properties bin when empty	When this is checked, the Properties Bin is closed when the last control panel in the Bin is closed. When this is NOT checked, the Properties Bin remains open when the last control panel in the Bin is closed.
expand/collapse panels in Properties bin to match selection	When this is checked, only the control panels of the nodes that are selected in the Node Graph are opened. All unselected nodes will have their control panels collapsed. This only applies to the control panels in the Properties Bin. When this is NOT checked, control panels are not opened or closed based on the selection in the Node Graph.
Input button does	Select what happens when you click the input button on top of a control panel and select one of the node's inputs: • **select input node only** - an input of the node is selected in the Node graph. • **scroll node into view** - an input of the node is selected in the Node Graph and, if the input is outside the currently displayed area of the Node Graph, the displayed area is adjusted to include the input node. • **center node** - an input of the node is selected and positioned in the center of the Node Graph.

Appearance Tab

Setting	Function
font	Change the type, weight, angle, and size of the font used on the application interface.
Colors	
Background	Change the background color of most user interface elements (menus, tool bars, panes, control panels, Viewers, and pop-up dialogs). To set the color back to default (dark gray), right-click on the button and select **Set color to default**.
Base	Change the color of input fields, the input pane of the Script Editor, and the left part of the Curve Editor. To set the color back to default (light gray), right-click on the button and select **Set color to default**.
Highlight	Change the color of the highlighting that appears when you hover the cursor over a control, select a file or folder in the File Browser, or scrub to a new frame on the timeline. To set the color back to default (light orange), right-click on the button and select **Set color to default**.

Setting	Function
Label	Change the color of labels and text on the application interface. Note that this does not set the color of the labels on the nodes in the Node Graph. To set the color back to default (white), right-click on the button and select **Set color to default**.
Button	Change the color of buttons and pulldown menus. To set the color back to default (medium gray), right-click on the button and select **Set color to default**.
Animated	Change the color that indicates a control has been animated. To set the color back to default (blue), right-click on the button and select **Set color to default**.
Keyframe	Change the color that indicates a key frame has been set. To set the color back to default (blue), right-click on the button and select **Set color to default**.
Playhead	Change the color of the frame marker on the Viewer timeline. To set the color back to default (orange), right-click on the button and select **Set color to default**.
In/Out Markers	Change the color of the in and out frame markers on the Viewer timeline. To set the color back to default (red), right-click on the button and select **Set color to default**.
File Browser Colors	
Queued Item Color	Change the color of items that are already selected for opening while the file brower dialog is still open.
Focus Color	Change the color of the highlighting that indicates that the mouse pointer is focused on an item.
File Browser Options	
Sequence Display Mode	Change the way sequences are displayed in the file browser dialog. Choose **Hashes (#)** to show hashes for the frame number buffers (for example, example.####.tif). Choose **Printf Notation (%d)** to mark them with printf notation (for example, example.%04d.tif)

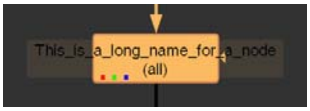
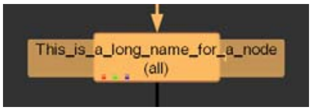
Node Colors Tab

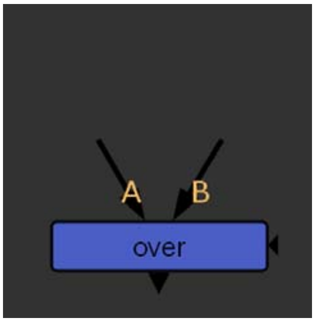
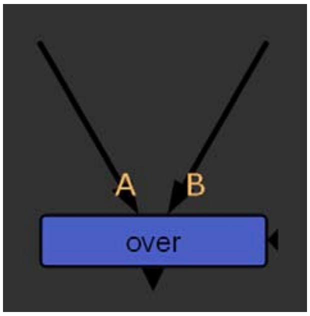
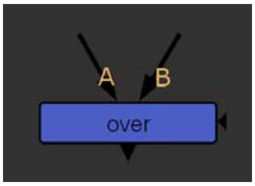
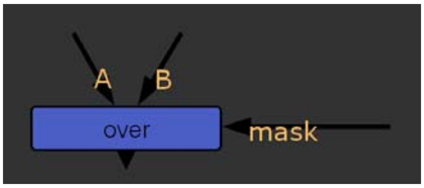
Setting	Function
autocolor	Check this to color nodes based on the class they belong to. If you don’t check this, the All Others color is used on all nodes.
Shade Nodes	Check this to have the nodes in the Node Graph shaded. If you don’t check this, no shading is used.

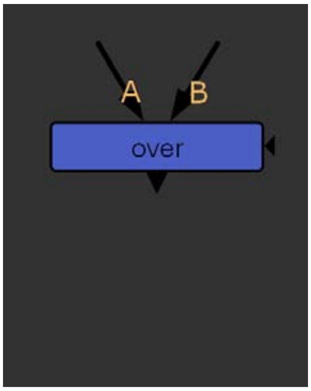
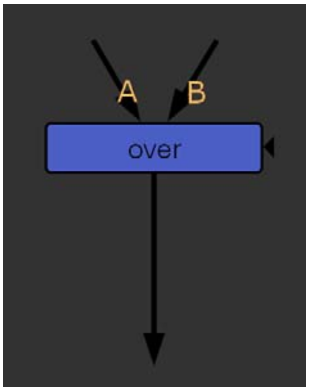
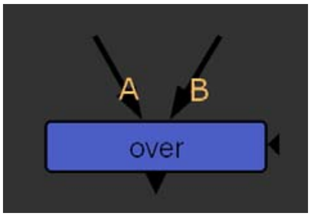
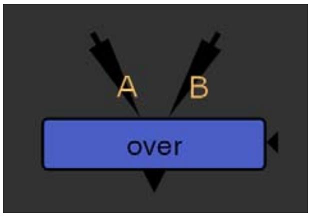
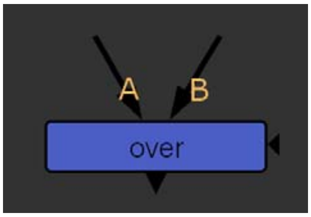
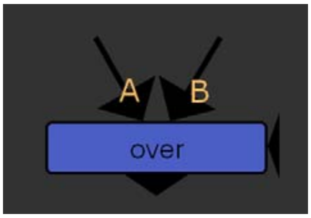
Setting	Function
RotoPaint, Drawing, Color, Time, Channel, Merge, Keyer, Write, 3D, Filters, 2D Transform	Change the color of the nodes that belong to each group. To set the color back to default, right-click on the color button on the right and select **Set color to default**. You can also change which nodes belong to each group by entering the names of the nodes in the text input fields.
User 1, User 2	You can use these controls to create a group of gizmos and change their color in the Node Graph. List the names of the gizmos (without the .gizmo extension) in the input field, and use the color button to change their color in the Node Graph. To set the color back to default (light gray), right-click on the color button on the right and select **Set color to default**.
All Others	Change the color of any nodes that do not belong to the node groups above. To set the color back to default (light gray), right-click on the button and select **Set color to default**.
Text	Change the color of the labels and text in the Node Graph (for example, the labels on nodes and the notes you create using StickyNote nodes). To set the color back to default (black), right-click on the button and select **Set color to default**.
Selected	Change the color that is used to indicate that nodes are selected. To set the color back to default (yellow), right-click on the button and select **Set color to default**.
Selected Input	Change the color that is used to display the input names of the nodes that are selected. To set the color back to default (off-white), right-click on the button and select **Set color to default**.
GL Color	Change the color of the Viewer overlays. To set the color back to default, right-click on the button and select **Set color to default**.

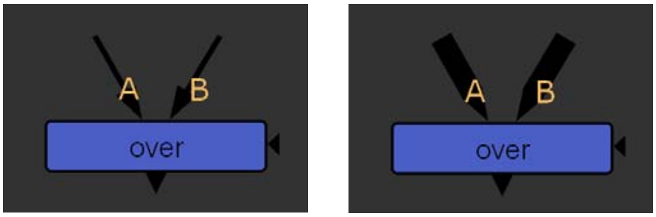
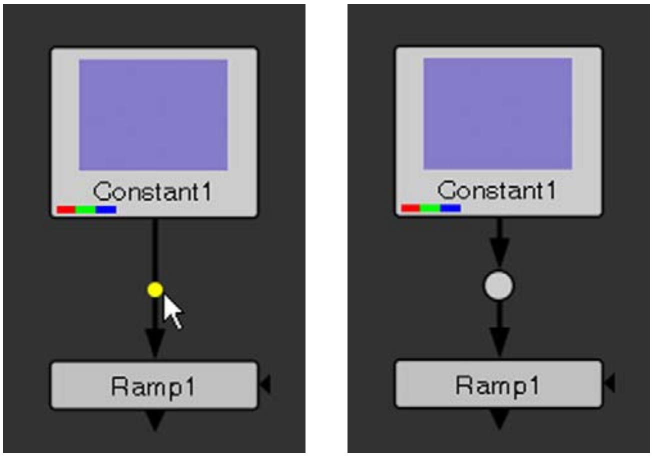
Node Graph Tab

Setting	Function
autolabel	Check this to automatically label nodes with channel/mask information. If you don't check this, nodes will only show the file name or the node name.
highlight running operators	Check this to highlight any nodes whose output is currently being calculated.

Setting	Function
node name background	When a node is selected and the node's name is too long to fit inside the node, a background is drawn behind the name to improve legibility. You can use this control to set the intensity of the background, from 0 (no background) to 1 (fully opaque background).  
label font	Change the type, weight, angle, and size of the font used on labels in the Node Graph.
tile size (WxH)	Change the width and height (in pixels) of the nodes in the Node Graph.
snap to node	When this is checked and you move a node in the Node Graph, the node snaps to a position that aligns it horizontally and vertically with its input and output nodes.
grid size (WxH)	Define the width and height (in pixels) of the cells in the background grid that you can display on the Node Graph. To see the grid, check **show grid** below.
snap to grid	When this is checked and you move a node in the Node Graph, the node snaps to a position that lines it up with the background grid lines. To see the grid, check **show grid** below.
show grid	Show a background grid on the Node Graph.
snap threshold	Define the maximum number of pixels to jump by when snapping nodes to other nodes or grid lines.
Colors	
Node Graph	Change the background color of the Node Graph. To set the color back to default (dark gray), right-click on the button and select **Set color to default**.
Overlay	Change the color of the grid that you can have appear on the Node Graph if you check **show grid** below. To set the color back to default (light gray), right-click on the button and select **Set color to default**.
Elbow	Change the color of the "elbows" that appear on arrows when you press **Ctrl/Cmd** on the Node Graph, and that you can click to insert Dot nodes. To set the color back to default (yellow), right-click on the button and select **Set color to default**.
Arrows	
Left arrow button	Change the color of arrows pointing left in the Node Graph. To set the color back to default (black), right-click on the button and select **Set color to default**.
Right arrow button	Change the color of arrows pointing right in the Node Graph. To set the color back to default (black), right-click on the button and select **Set color to default**.

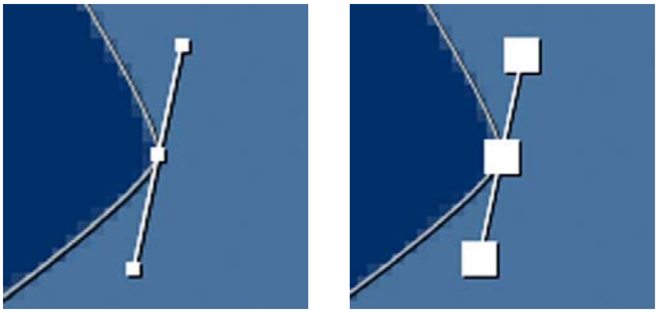
Setting	Function
Up arrow button	Change the color of arrows pointing up in the Node Graph. To set the color back to default (black), right-click on the button and select **Set color to default**.
Down arrow button	Change the color of arrows pointing down in the Node Graph. To set the color back to default (black), right-click on the button and select **Set color to default**.
expression arrows	Change the color of the arrows that indicate nodes are connected via an expression. To set the color back to default (green), right-click on the button and select **Set color to default**.
enable	Check this if you want to display expression arrows in the Node Graph.
clone arrows	Change the color of the arrows that indicate nodes have been cloned. To set the color back to default (orange), right-click on the button and select **Set color to default**.
enable	Check this if you want to display clone arrows in the Node Graph.
unconnected top input arrow length	Adjust the length of the unconnected input arrows that appear on top of nodes in the Node Graph. By default, this value is set to 35 pixels. The maximum value is 70.  
unconnected left/right input arrow length	Adjust the length of the unconnected arrows that appear on the sides of some nodes in the Node Graph. This affects any mask inputs and extra Viewer or Scene node inputs, for example. By default, this value is set to 4 pixels. The maximum value is 70.  

Setting	Function
unconnected output arrow length	Adjust the length of unconnected output arrows in the Node Graph. By default, this value is set to 8 pixels. The maximum value is 70.  
arrow head length	Adjust the length of arrow heads in the Node Graph. By default, this value is set to 12 pixels. There is no maximum value.  
arrow head width	Adjust the width of arrow heads in the Node Graph. By default, this value is set to 8 pixels. There is no maximum value.  

Setting	Function
arrow width	Adjust the width of the arrows in the Node Graph. By default, this value is set to 2 pixels. There is no maximum value. 
allow picking of connected arrow heads	When this is checked, you can click on an arrow head and drag it to a new location. When this is NOT checked, connected arrow heads are locked into place, and you can only change the connections by moving the arrow tails.
allow picking of arrow elbows to create Dots	When this is checked, you can press **Ctrl** (**Cmd** on a Mac) on the Node Graph to display yellow “elbows” on the Node Graph arrows and click on these to insert Dot nodes.  If you **Ctrl/Cmd+Shift+click** on an elbow, the new Dot node is branched off to a new arrow rather than inserted in the existing arrow. When this setting is NOT checked, adding Dot nodes in this manner is not possible.
drag-to-insert only works near middle of arrows	When this is NOT checked, you can insert nodes in between other nodes by dragging them over any point of the connecting arrow. When this is checked, you can only insert nodes in between other nodes in the above manner by dragging them over the middle point of the connecting arrow.
size of dots	Adjust the size of Dot nodes. By default, the value is set to 1.

Viewers Tab

Setting	Function
new Viewers go to own window	When this is checked, each new Viewer you create appears in its own floating window. When this is NOT checked, additional Viewers will attempt to dock with existing Viewers.
delete Viewer node when Viewer window is closed	When this is checked, Viewer nodes are deleted from the Node Graph when you close the associated Viewers in the Viewer pane. When this is NOT checked, closing Viewers in the Viewer pane does not affect Viewer nodes in the Node Graph.
apply LUT to color channels only	When this is checked, look-up tables (LUTs) are only applied to the red, green, and blue channels. When this is NOT checked, LUTs are applied to all channels.
zoom lock on for new viewers	When this is checked, new Viewers for inputs of different sizes maintain the current zoom level rather than the current on-screen image dimensions. When this is NOT checked (default), new Viewers will automatically zoom the input to match the current on-screen image dimensions.
viewer buffer bit depth	Set the default Viewer OpenGL buffer depth: • **float** - Uses a full float texture. • **half-float** - Converts to 16-bit (half) float. • **byte** - Uses 8-bit with error diffusion. This is the default. When Nuke is set to use 8-bit (**byte**) **gl buffer depth**, we perform error diffusion in the conversion from the internal floating point image data to 8-bit. This is to avoid the appearance of banding due to the reduced bit depth. Without this, artists often mistakenly assume the banding they see is actually present in their image data and try to blur to remove it, softening the image unnecessarily. However, because Nuke works internally at floating point, there is little likelihood of banding actually being present. When **gl buffer depth** is set to **half-float** or **float**, an ordered dither is applied, rather than error diffusion.
use GPU for viewer when possible	When this is checked, the Viewer applies its effects (such as gain, gamma, and the Viewer Process node) in the GPU when possible. However, in some cases, like when monitor output is enabled or **gl buffer depth** is set to **byte** in the Viewer settings, effects must still be computed in the CPU.
use GPU for inputs when possible	Normally, the Viewer only attempts to run its own effects (such as gain, gamma, and the Viewer Process node) on the GPU. However, when this is checked, any nodes connected to the Viewer are also computed on the GPU when possible. Note that this cannot be done for all nodes because not all nodes have a GPU implementation. If nodes are computed on the GPU, the color values displayed in the Viewer will be inaccurate. This is because they show the color from the last node computed in the CPU prior to transferring the image into the graphics card.

Setting	**Function**
texture size	Select the size of the texture maps for the OpenGL preview of 3D objects and 2D transformations. The default size is 512x512.
2D bg	Change the background color of the 2D Viewer. To set the color back to default (black), right-click on the button and select **Set color to default**.
2D fg	Change the color of borders and text in the 2D Viewer. To set the color back to default (light gray), right-click on the button and select **Set color to default**.
3D bg	Change the background color of the 3D Viewer. To set the color back to default (black), right-click on the button and select **Set color to default**.
3D fg	Change the color of borders and text in the 3D Viewer. To set the color back to default (light gray), right-click on the button and select **Set color to default**.
Interaction	
middle button pans	Check this to use the middle mouse button to pan in the Viewer, Node Graph and the Curve Editor.
left+middle to zoom	Check this to use the left and the middle mouse button together to zoom in the Viewer, Node Graph and the Curve Editor.
3D control type	Select the navigation control scheme you want to use in the 3D Viewer: Nuke, Maya, Houdini, or Lightwave.
handle size	Adjust the size of the square control handles that appear on the Viewer for some operations, such as transformations, warps, and Bezier and B-spline shapes. By default, this value is set to 5. 
handle pick size	Adjust the size of the pickable area for handle points, such as those found on RotoPaint shapes. For example, If the value is 20, you can pick up the point (for instance, when click-dragging) within a 20 pixel's radius from the point. By default, this value is set to 5.
icon size	Adjust the size of the 2D transformation overlay, 3D camera, 3D object normals, and 3D axis on the Viewer. By default, this value is set to 50.

Setting	Function
icon scaling	Adjust how much the scale of display affects the size of the 2D transformation overlay, 3D camera, and 3D axis. When this is set to 0, these controls are always drawn the same size, regardless of the zoom level. When the value is set to 1, the controls scale with the displayed image or 3D scene when you zoom in or out. Intermediate values mix this so that the controls do scale, but not as much as the image does. This gives an optical illusion that you are zooming in or out without making the controls unusably small or large.
object interaction speed	Set how fast mouse movements rotate and translate 3D axis and cameras. The lower the value, the finer the movements. The default value is 0.1.
camera interaction speed	Set how fast mouse movements tumble and roll the 3D view in the Viewer. The lower the value, the finer the movements. The default value is 1.
Roto Splines	
Points	Change the default color of the points on RotoPaint shapes and strokes. To set the color back to default (gray), right-click on the button and select **Set color to default**.
Curves	Change the default color of the RotoPaint shape and stroke curves. To set the color back to default (gray), right-click on the button and select **Set color to default**.
Transform	Change the default color of the RotoPaint transform jack. To set the color back to default (gray), right-click on the button and select **Set color to default**.
Line Width	Adjust the width of lines in the RotoPaint strokes and shapes.
Draw Shadow	When this is checked, a shadow is drawn for lines in the RotoPaint shapes and strokes. This can make the lines easier to see. When this is NOT checked, no shadows are drawn.

Script Editor Tab

Setting	Function
font	Change the type, weight, angle and size of the font used in the Script Editor.
Input	
clear input window on successful script execution	When this is checked, any successfully executed Python statements disappear from the input pane of the Script Editor and appear in the output pane. When this is NOT checked, all statements stay in the input pane of the Script Editor, even if they were successfully executed.

Setting	Function
echo python commands to output window	Check this to have all Python commands executed by yourself or Nuke appear in the output pane of the Script Editor. This way, you can, for example, select a node from the Toolbar and have the corresponding Python command displayed in the output pane.
Highlighting	
keywords	Change the color that's used to highlight any words that are Python's keywords (for example, print and import) in the Script Editor. To set the color back to default (green), right-click on the button and select **Set color to default**.
string literals (double quotes)	Change the color that's used to highlight strings (inside double quotation marks) in the Script Editor. To set the color back to default (red), right-click on the button and select **Set color to default.**
string literals (single quotes)	Change the color that's used to highlight strings (inside single quotation marks) in the Script Editor. To set the color back to default (cyan), right-click on the button and select **Set color to default.**
comments	Change the color that's used to highlight comments (anything beginning with #) in the Script Editor. To set the color back to default (yellow), right-click on the button and select **Set color to default.**

20 THE SCRIPT EDITOR AND PYTHON

Have you ever thought of something you'd absolutely love to be able to do with Nuke but that does not seem to have been a priority for Nuke developers? Or would you just die to be able to automate some of the more tedious procedures, so that they'd take care of themselves while you aren't even at your desk? Well, we have just the answer for you: Python, one of the two scripting languages Nuke supports (the other is TCL).

This chapter describes how you can use the Script Editor for executing Python scripts, gives you examples of those scripts, teaches you how to automate your scripts, and directs you to sources of more information.

Workflow

The workflow for using Python in Nuke is generally the following:

1. Enter Python statements in Nuke's Script Editor to perform the actions you want to perform.
2. Save your script with the extension .py in a directory that is contained in the sys.path variable.
3. Later, when you want to execute the same statement sequence, import the .py file into Nuke's Script Editor. Nuke executes the statements in the specified order.

Note *You can also run an interactive Python session on the command line with **nuke -t**. However, on Mac OS X the Python interpreter and other applications that are bundled with Nuke (such as IDLE) can only be run from inside the directory where they are located.*

Importing the Nuke Python module outside of Nuke is not possible on any platform.

Note *You should not give your Nuke Python modules a name that's already used by an existing Nuke node (for example, Transform). If you do so, importing the module will fail. This is because when Python searches for the module, it finds a .so file with the same name (the node itself) and tries to import that. If this is the case, you get an error message saying dynamic module does not define init function.*

Note *There is a known issue with the build of Python 2.6.5 in Nuke 6.2 on Linux 64 only. In order to import urllib2 (useful for communication with web services and HTTP servers), the openssl and crypto libs must be installed.*

The build of Python 2.6.5 also expects to be able to link against libssl.so.4, so you may need to create a symbolic link to libssl.so from libssl.so.4.

Example: ln -s libssl.so libssl.so.4

The urllib2 module is dynamically loaded in Python, and will fail to load if it can not find libssl.so.4 or libcrypto.so. If this is the case, you will most likely see an error message similar to this:

```
Traceback (most recent call last):
File "<string>", line 2, in<module>
File "/usr/local/Nuke[version number]/lib/python2.6/urllib2.py", line 93, in<module>
import hashlib
File "/usr/local/Nuke[version number]/lib/python2.6/hashlib.py", line 136, in<module>
md5 = __get_builtin_constructor('md5')
File "/usr/local/Nuke[version number]/lib/python2.6/hashlib.py", line 63, in
__get_builtin_constructor
import _md5
ImportError: No module named _md5
```

Using the Script Editor

You type Python scripts into Nuke's Script Editor.

To open the Script Editor:

To open the Script Editor, click on one of the content menus and select **Script Editor** from the menu that opens.

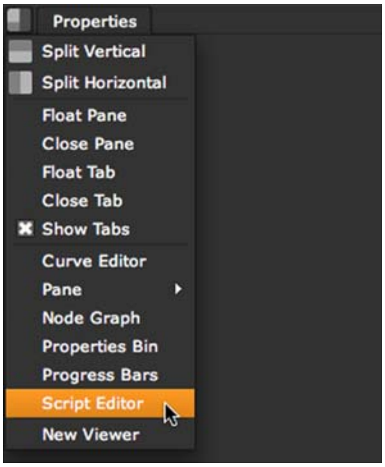
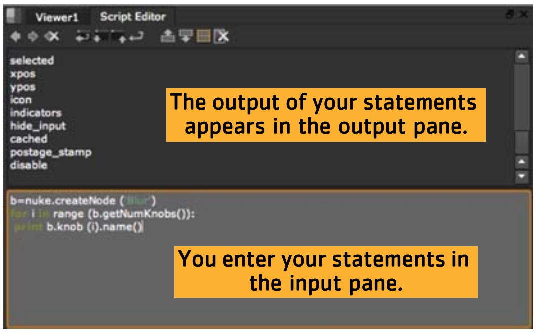

Properties
Split Vertical
Split Horizontal
Float Pane
Close Pane
Float Tab
Close Tab
Show Tabs
Curve Editor
Pane
Node Graph
Properties Bin
Progress Bars
Script Editor
New Viewer

Figure 20.1: Opening the Script Editor.

Input and output panes

The Script Editor is divided into two parts, as shown in Figure 20.2. You use the lower part (input pane) to type in and execute your Python statement, and when you have done so, statements and their outputs appear in the upper part of the editor (output pane). Successfully executed statements are followed by a hash mark (#).


Viewer1
Script Editor

```
selected
xpos
ypos
icon
indicators
hide_input
cached
postage_stamp
disable
```

The output of your statements appears in the output pane.

```
b=nuke.createNode ('Blur')
for i in range (b.getNumKnobs()):
 print b.knob (i).name()
```

You enter your statements in the input pane.

Figure 20.2: The two parts of the Script Editor.

To hide the output or input pane, click the **Show input only** or **Show output only** button on top of the Script Editor.

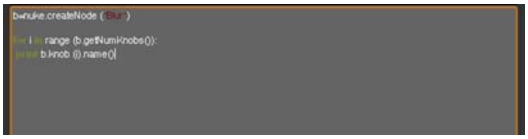
To show both panes again, click the **Show both input and output** button.

To enter a statement:

1. Click on the input pane of the editor to insert the cursor there.

2. Type in your statement. To use the usual editing functions, such as copy and paste, right-click on the editor and select the desired function.

   When entering the statement, you'll notice that any words that are Python's keywords (such as *print* and *import*) turn green, while strings (basically, anything in quotation marks) become either red or cyan. Comments are shown in yellow.



```
b=nuke.createNode ("Blur")

for i in range (b.getNumKnobs()):
 print b.knob (i).name()
```

   If you like, you can change these colors and the font on the **Script Editor** tab of the Preferences dialog. To open the preferences, press Shift+S.

Tip *You can also use auto-complete to help you with entering Python statements. Start writing a command and press the* ***Tab*** *key. If there's only one way to end your command, Nuke will autocomplete it straight away. If there are several possible completions, Nuke will give you a pop-up menu listing them. If there's no known way to complete your command, nothing will happen. Even if your command is automatically completed, it will not be executed automatically, just in case you don't like surprise side effects.*

3. If your statement includes several lines or you want to enter several statements at once, press **Return** to move to the next line.
4. To execute the statement, click the **Run the current script** button on the top of the Editor, or press Ctrl+Return (Cmd+Return on a Mac).

Tip *You can also execute statements by pressing Ctrl+Enter (Cmd+Enter on a Mac) on the numeric keypad.*

By default, successful statements disappear from the input pane, and appear in the output pane. However, if you want all statements to stay in the input pane after they are executed, you can do the following:

1. Press Shift+S to open the Preferences dialog.
2. Go to the **Script Editor** tab.
3. Uncheck **clear input window on successful script execution**.
4. Click **Close** to save the preference for the current project only, or **Save Prefs** to save the preference for the current and future projects.

If you enter an invalid statement, Nuke produces an error in the output pane of the Script Editor, leaving the invalid statement in the input pane.

Correct the statement and execute it again until you get it right.

Note *Sometimes you may get an error if you copy and paste statements into the Script Editor from another source, like an e-mail. This may be caused by the mark-up or encoding of the source you copied the statement from. To fix the problem, re-enter the statement manually.*

If you want to have all executed Python commands appear in the output pane of the Script Editor, open the Preferences dialog (press Shift+S), go to the **Script Editor** tab, and check **echo all commands to output window**. This applies to both commands executed by yourself and by Nuke. For example, if you select a node from the Toolbar, the corresponding Python command is displayed in the output pane. This does not apply to all actions you take in the graphical user interface, however, but only those that are performed by executing Python script commands.

To only execute part of a script, enter the script in the input pane and select the part you want to execute. Press Ctrl+Return (Cmd+Return on a Mac). Nuke runs the selected part of the script, leaving the script in the input pane.

To repeat a statement, click the **Previous Script** button on top of the Editor to move back to the previous statement. You can do this several times until you reach the statement you want to repeat. To execute the statement again, press Ctrl+Enter (Cmd+Enter on a Mac).

To increase the indentation in the input window, press Tab.

To decrease the indentation in the input window, press Shift+Tab.

To move through or clear the script history:

In addition to stepping backwards through the history of your script, you can also step forwards. Click the **Next script** button to move forward through your statements.

To clear the history, click the **Clear history** button.

To clear the output pane:

Click the **Clear output window** button (or press Ctrl/Cmd+Backspace).

So, now you know how to use the editor. But what do you do with this knowledge if you don't know any statements? Not much. Next, we look at some example scripts you can enter in the editor.

Example Scripts

As you will soon notice after trying to enter a few scripts yourself, the scripts are case-sensitive and will only work if you enter them exactly right. There's a bit of flexibility when it comes to quotation marks and spaces, though. You can use both single and double quotes in your statements, and you don't necessarily need to add a space before the brackets like we have done in the following examples.

Note *If you copy and paste example scripts from this user guide into a text editor, line indentations may not be preserved. If this is the case, correct the indentations manually.*

Creating Nodes and Setting Their Parameters

This section explains how to create nodes and set their parameters using Python.

To create a node, enter one of the following:

- **nuke.nodes.*nodename* (...)**

  where *nodename* is the name of the node you want to create. For example, to add a Blur node, enter **nuke.nodes.Blur ()**.

  You can only use this statement if you know the name of the node you want to create and can enter it explicitly in the statement.

  Note that when you have a node selected and create a node this way, the new node will NOT automatically be connected to the selected node. If you want the node to be connected, you can use the **setInput()** function, for instance, **nuke.nodes.Blur().setInput(0, nuke.selectedNode())**. This will also work in earlier versions of Nuke. Alternatively, you can use the **nuke.createNode (...)** statement described below.

  When using the **nuke.nodes.*nodename* (...)** statement to create nodes, you can add optional named arguments inside the parenthesis, separated by commas. You can use these arguments to set parameter values for the node, rename the node, set the nodes inputs, and (in the case of Read nodes) specify where to load an image from. Setting parameter values, renaming nodes, and creating Read nodes are described below, whereas setting a node's inputs is discussed later in this chapter, under "Connecting Nodes and Setting Their Inputs" on page 570.
- **nuke.createNode (...)**

  If you wanted to create a Blur node using this statement, you would enter **nuke.createNode ("Blur")**.

Note that when you have a node selected and create a node this way, the new node will automatically be connected to the selected node.

Note *You may encounter problems if you attempt to use the* ***nuke.nodes.nodename (...)*** *or* ***nuke.createNode (...)*** *statements to call a file that does exist in your [Nuke directory]/plugins folder but is NOT normally used to create a node or called directly. For example, entering* ***nuke.nodes.drop ()*** *or* ***nuke.createNode("drop")*** *causes Nuke to try and call the drop.tcl file, but because no node called Drop exists, this may result in a crash.*

In general, any files in the plugins directory whose name does not start with a capital letter do not create nodes and should not be used with the ***nuke.nodes.nodename*** *or* ***nuke.createNode*** *statements.*

To create a node and set a parameter value, enter:

nuke.nodes.*nodename* (*parameter*=*value*)

where

- *nodename* is the name of the node you want to add
- *parameter* is the name of the control whose value you want to set
- *value* is the value you want to give to the control.

For example, if you wanted to create a Blur node and set its size parameter to 10, you would need to enter nuke.nodes.Blur (size=10).

You can check the results in the Blur node's properties panel, where the value in the **size** field should be set to 10.

To create a node and rename it, enter:

nuke.nodes.*nodename* (name="*newname*")

where

- *nodename* is the name of the node you want to add
- *newname* is the name that you want to give to the node.

For example, if you wanted to create a Camera node but call it *Projection_Cam*, you would enter nuke.nodes.Camera (name="Projection_Cam").

To create a Read node, enter:

nuke.nodes.Read (file="*filepath\filename.ext*") where *filepath\filename.ext* represents the path to the image you want to read in. For example, if you wanted to read in myimage.cin from C:\Temp, you would enter `nuke.nodes.Read (file="C:\Temp\myimage.cin")`.

Assigning Variables

Since you are likely to want to play with the node after creating it, it makes sense to assign the node to a variable. Then, in your subsequent statements, you can use the variable to refer to the node.

To add a node and assign it to a variable, enter:

```
variable=nuke.nodes.nodename (...)
```

where *variable* represents any character or string of characters you want to use as the variable, and *nodename* the name of the node you want to create.

For example, you could enter:

```
b=nuke.nodes.Blur ()
```

Here, you are adding a Blur node and deciding to call it b in your subsequent statements. Of course, you could call it anything you like. If you were adding several Blur nodes to your script, you might want to call them b1, b2, and b3, for example.

Now, say you created a Blur node and then decided to set its **size** parameter to 10. If you have assigned the node to a variable (for example, b), you can do this by entering `b["size"].setValue (10)`.

Adding Parameters to Nodes

We've now looked at setting parameter values for the existing parameters of a node. But what should you do if you want to add a completely new parameter to a node? You need to use, for example, the following statements:

```
b = nuke.nodes.nodename (...)
k = nuke.Array_Knob("name", "label")
b.addKnob(k)
```

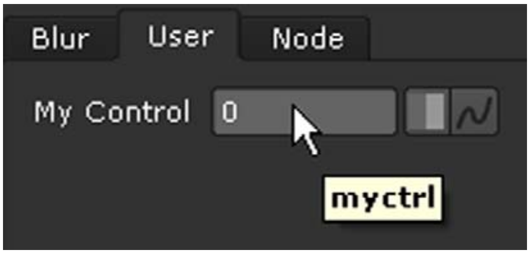
where

- *nodename* represents the name of the node you want to add the new parameter to.
- *name* represents a unique name you give the parameter to reference it using scripts and expressions.
- *label* represents the label shown next to the control in the node's properties panel.

Let's say you enter the following lines:

```
b = nuke.nodes.Blur ()
k = nuke.Array_Knob("myctrl", "My Control")
b.addKnob(k)
```

Using these statements, you add a Blur node and assign it to the variable b. You then create an input field parameter, decide to call it "myctrl" in references and "My Control" on the user interface, and assign the parameter to the variable k. Finally, you add the parameter (k) to the Blur node's (b) controls.



If you wanted to create a slider control rather than an input field, you'd need to use the following statements (replacing **Array_Knob** with **WH_Knob**):

```
b = nuke.nodes.Blur ()
k = nuke.WH_Knob("myctrl", "My Control")
b.addKnob(k)
```

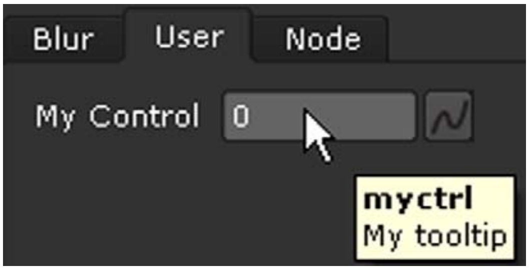
The following lines, instead, would produce a checkbox control:

```
b = nuke.nodes.Blur ()
k = nuke.Boolean_Knob("myctrl", "My Control")
b.addKnob(k)
```

Now you have a control, but it has no tool tip. Let's add one. Enter the following statement:

```
k.setTooltip('My tooltip')
```

This adds a tool tip with the words *My tooltip* to the control created earlier and assigned to variable k.



Note *To see the tool tip, you may need to close and reopen the node's properties panel.*

To show or hide a node's control panel, enter:

You can use the **showControlPanel()** and **hideControlPanel()** functions to set whether you want the control panel of a node to be open or not. For example you can enter

n = nuke.toNode("Blur1")
n.showControlPanel()

to show the control panel of a new Blur node.

Connecting Nodes and Setting Their Inputs

You can use a script to set an input for a node.

Let's imagine you want to add two Read nodes and a Merge node (in this case, an over node) to your node tree, and connect the Read nodes to the over node's **A** and **B** inputs (numbers 1 and 0), like this:

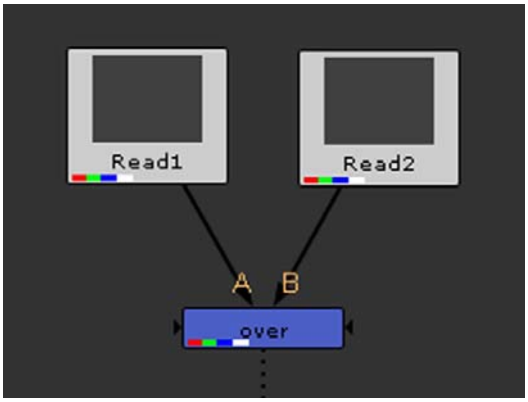


Figure 20.3: Read nodes and a Merge (over) node.

You need to add the Read nodes and the over node and specify the over node's inputs using, for example, the following statements:

- r1=nuke.nodes.Read (file="*filepath\filename.ext*")
- r2=nuke.nodes.Read (file="*filepath\filename.ext*")
- m=nuke.nodes.Merge (inputs=[r2, r1])

where *filepath\filename.ext* represents the locations and names of the images you want to read in. The last statement creates a Merge node and sets r2 (the second Read node) and r1 (the first Read node) as its inputs.

Setting Default Values for Controls

You can set default values for controls in any nodes that belong to the same class. After a default value is set, all controls with matching names will default to this value. To set a default value, use the following statement:
`nuke.knobDefault()`

For example, if you want the **size** control on Blur nodes to default to 20, you can use the following statement:
`nuke.knobDefault("Blur.size", "20")`

If you then wanted the last frame value of the **frame range** controls in the Project Settings to default to frame 200, you could use the following statement:
`nuke.knobDefault("Root.last_frame", "200")`

Notice that you need to start the node class with a capital letter (for example, Root rather than root).

It is also important that you add these statements in your init.py file rather than menu.py. This ensures that they are set for command line start-up as well as the graphical user interface (GUI). If these statements are not processed until menu.py (GUI start-up), then nodes created from the command line Nuke Python prompt (that is, starting as nuke -t) will not have the defaults applied.

The init.py file is a file that is run before menu.py usually to set things that are GUI independent. It should be located in your plug-in path directory (for more information on plug-in path directories, see "Environment Variables" on page 611). If the init.py file does not yet exist, simply create a text file and name it init.py.

Rendering with the Write Node

Let's imagine you've added a Write1 node into your script. Now, you want to render every other frame of the frames from 1 to 35.

To render a single Write node, enter:

nuke.execute (*"name",start,end,incr*)

where

- **name** represents the name of the write node, for example **Write1**
- **start** represents the first frame you want to render
- **end** represents the last frame you want to render
- **incr** represents the increment you want to use when rendering. For example, if you used 3 here, Nuke would render every third frame.

In our example, you would enter
nuke.execute ("Write1",1,35,2)
or
nuke.execute ("Write1",start=1,end=35,incr=2)

Tip *Instead of **nuke.execute (name,start,end,incr)**, you can also use **nuke.render (name,start,end,incr)**. Both statements perform the same action.*

To render multiple Write nodes or ranges, enter:

nuke.executeMultiple ((*variable*,), ([*start,end,incr*],))

where

- **variable** represents the variables of the write nodes
- **start** represents the first frames you want to render
- **end** represents the last frames you want to render
- **incr** represents the increments you want to use when rendering. For example, if you used 3 here, Nuke would render every third frame.

Flipbooking with External Applications

FrameCycler is Nuke's default application for flipbooking but you can also use other applications for flipbooking. In order to do this you must implement a Python class. An example on how to do this based on RV, which is also used in the flipbooking application example in the pyQtExamples directory. To use this, do the following:

1. Locate the flipbookingExample.py and edit it to your circumstances as it won't work straight away.
2. Put the modified Python file in your .nuke directory, saving it for example as myflipbook.py.
3. In your init.py (or menu.py) file, add the following line:
   from myflipbook import *

Refer to the nukescripts.flipbooking.FlipbookApplication module in the Python Scripting documentation for further details.

Listing a Node's Controls

Let's try something a little more complicated and list all the controls in a node's properties panel.

To list a node's controls, enter:

```
for i in range (getNumKnobs()):
 print knob (i).name()
```

Make sure you enter this compound statement on two separate lines and indent the second line.

For example, to list the controls of the Blur node you added earlier and assigned to variable **b**, you would enter:

```
for i in range (b.getNumKnobs()):
 print b.knob (i).name()
```

As a result of this, Nuke lists the Blur node's controls, displaying them in the output pane of the Script Editor.

Undoing and Redoing Actions

To undo actions, enter:

```
nuke.undo()
```

To redo actions, enter:

```
nuke.redo()
```

Frame Navigation

To use the frame navigation buttons of the currently active Viewer, enter the following statement:

```
nuke.activeViewer().frameControl(i)
```

where **i** is an integer that indicates the frame navigation button you want to execute. You can replace **i** with the following:

- **-6** to go to the first frame.
- **-5** to play the sequence backward.
- **-4** to go to the previous keyframe.
- **-3** to step back by increment.
- **-2** to go back to the previous keyframe or increment, whichever is closer.

- **-1** to step back one frame.
- **0** to stop playback.
- **+1** to step forward one frame.
- **+2** to go to the next keyframe or increment, whichever is closer.
- **+3** to step forward by increment.
- **+4** to go to the next keyframe.
- **+5** to play the sequence forward.
- **+6** to go to the last frame.

You can also assign these buttons to a hotkey. For example, to assign the play button to the Up arrow key, do the following:

1. Create a file called `menu.py` in your plug-in path directory if one doesn't already exist.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Add the following entry in your menu.py:

   ```
   def play():
     v = nuke.activeViewer()
     if v:
       v.play( 1 )


   menubar = nuke.menu("Nuke");
   m = menubar.addMenu("&File")
   m.addCommand("@;Play", "play()", "Up")
   ```

This assigns the hotkey to an invisible menu item (**@** before the name of the menu item makes the item invisible).

Setting Frame Ranges Using Python

Setting a single frame range

To set a single frame range, use:

```
nuke.FrameRange()
```

This can only be used to store *one* frame range. You can set the frame range using, for example:

```
frange = nuke.FrameRange("1-100×2")
```

The **1-100x2** syntax in the above statement sets the frame range to 1, 3, 5, 7, 9, 11...99. For more examples of the syntax that you can use, see "Defining Frame Ranges" on page 140.

If the frame range is not valid, an exception will be thrown.

To iterate all frames in the frame range you defined above, you can use:

```
for f in frange:
  print f
```

You can also use the following methods:

- **frange.setFirst(int)** to set the first frame.
- **frange.setLast(int)** to set the last frame.
- **frange.setIncrement(int)** to set the increment.
- **frange.first()** to get the first frame.
- **frange.last()** to get the last frame.
- **frange.increment()** to get the increment.
- **frange.frames()** to get the number of frames in the frame range.
- **frange.getFrame(int)** to get the frame at a specific index.
- **frange.isInRange(int)** to return True if a frame is in the frame range, or False if it's not.
- **frange.minFrame()** to return the smallest frame number in the frame range.
- **frange.maxFrame()** to return the largest frame number in the frame range.
- **frange.stepFrame()** to return the absolute increment between one frame and another. For example, if the real increment in the frame range was -3, this would return 3.

Storing multiple frame ranges

To store multiple frame ranges, use:
nuke.FrameRanges()

When using nuke.FrameRanges(), you can define the frame ranges in the following ways:

- by listing the frames to include, for example:
  franges = nuke.FrameRanges([1, 2, 3, 6, 7, 8])
- by using a single string, for example:
  franges = nuke.FrameRanges("1-100 2-300x2")
- by using multiple strings, for example:
  franges = nuke.FrameRanges(["1-10x1", "15-18x1 30-40x2"])
- by using multiple nuke.FrameRange() calls, for example:
  franges = nuke.FrameRanges([nuke.FrameRange(1,100,5), nuke.FrameRange(200,250,3)])

For more examples of the syntax that you can use when defining frame ranges in Nuke, see "Defining Frame Ranges" on page 140.

To iterate all the frame ranges stored (in this case, using the **franges** variable), you can use:

```
for r in franges:
  for f in r:
    print f
```

This prints all the frames stored in the frame ranges.

You can also use the following methods:

- **franges.size()** to return the number of frame ranges stored.
- **franges.add()** to add another frame range to the stored frame ranges.
- **franges.minFrame()** to get the smallest frame number in the stored frame ranges.
- **franges.maxFrame()** to get the largest frame number in the stored frame ranges.
- **franges.clear()** to reset all the stored frame ranges.
- **franges.compact()** to optimize the way the frame ranges are expressed, removing any duplicates.
- **franges.toFrameList()** to list all frames in the stored frame ranges.
- **franges.getRange()** to get a frame range in the stored frame ranges.

The **franges.compact()** method can be used to optimize the way the stored frame ranges are expressed. Where possible, it removes duplicate frame ranges and expresses the stored frame ranges in a more compact way. Here are two examples:

Example 1:

```
franges1 = nuke.FrameRanges("10-7x-1 1-5×1 3-7×1")
franges1.compact()
print "Ranges1: " + franges
```

This returns the following:

```
Ranges1: 1-10×1
```

Example 2:

```
franges2 = nuke.FrameRanges("10-7x-1 6-8×1 1-4×1 2-3×1")
franges2.compact()
print "Ranges2: " + franges2
```

This returns the following:

Ranges2: 1-4x1 6-10x1

Marking frame numbers in file names

It's possible to mark frame numbers in file names both with hashes (#) and in the printf (%04d) style. This is why you may find it useful to use **nukescripts.replaceHashes()** function to replace a sequence of hashes with the equivalent printf style number formatting. This makes it easy to do filename frame number substitutions like this:

```
filename = nukescripts.replaceHashes( node['file'].value() ) % nuke.frame()
```

Copying an Animation Curve Between Nodes

To copy an animation curve from one node to another, for example, from Blur1 to Blur2, you can do the following:

```
# The following statements create the two Blur nodes (Blur1 and Blur2) and assign them to variables b1 and b2.
b1 = nuke.nodes.Blur()
b2 = nuke.nodes.Blur()
# The following assigns the size control of the Blur1 node to the variable k1.
k1 = b1['size']
# The following three statements animate the control assigned to k1. At frame 30, its value is set to 10. At frame 40, it's set to 20.
k1.setAnimated()
k1.setValue(10, time = 30)
k1.setValue(20, time = 40)
# The following assigns the size control of the Blur2 node to the variable k2.
k2 = b2['size']
# The following copies the animation curve from k1 (size on Blur1) to k2 (size on Blur2).
k2.copyAnimations(k1.animations())
```

Overriding the Creation of a Particular Node

Sometimes, you may want to override the creation of a particular node. For example, Nuke includes two versions of the Merge node: Merge and Merge2. By default, Merge2 is inserted when you select **Merge** > **Merge**. However, if you'd prefer to use Merge, you can override the creation of the Merge node so that Nuke creates the Merge node by default. To do so, follow these steps:

1. Create a file called **menu.py** in your plug-in path directory if one doesn't already exist.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. Add the following entry:

```
class MyCustomNodes():
  def _getattr_(self, args):
    if args == "Merge2": args = "Merge"
    return nuke.NodeConstructor(args)
nuke.nodes = MyCustomNodes()
```

Tip *You can also do the same using the **nuke.createNode** statement rather than the **nuke.nodes.nodename** statement. In this case, add the following entry instead of the entry described in step 2:*

```
def createMyCustomNodes(node, knobs = "", inpanel = True):
  if node == "Merge2": node = "Merge"
  return nukeOriginalCreateNode(node = node, knobs = knobs, inpanel = inpanel)
nuke.createNode = createMyCustomNodes
```

Getting Information on the Nuke Environment You Are Running

In the Nuke module, there is an object called env. This object gives you information about the particular Nuke environment you are running. You can access it in a dictionary form: nuke.env["key"].

Currently, you can use the following statements to get the following information:

- nuke.env ["PluginExtension"]. This returns the file extension for plug-ins. The extension can be .dll on Windows, .so on Linux, and .dylib on Mac.
- nuke.env ["NukeVersionMajor"]. This returns the first number in the version number of Nuke. For example, if you were running Nuke 6.2v1, this would return 6.
- nuke.env ["NukeVersionMinor"]. This returns the second number in the version number of Nuke. For example, if you were running Nuke 6.2v1, this would return 1.
- nuke.env ["NukeVersionRelease"]. This returns the number after the letter v in the version number of Nuke. For example, if you were running Nuke 6.2v1, this would return 2.
- nuke.env ["NukeVersionPhase"]. This returns the last string in the version number of a Nuke beta. For example, if you were running Nuke 6.2v1b3, this would return b.
- nuke.env ["NukeVersionPhaseNumber"]. This returns the last number in the version number of a Nuke beta. For example, if you were running Nuke 6.2v1b3, this would return 3.

- **nuke.env ["NukeVersionDate"]**. This returns the date the version of Nuke you are running was built.
- **nuke.env ["NukeVersionString"]**. This returns the entire version number of Nuke.
- **nuke.env ["threads"]**. This returns the number of threads that will be spawned when rendering.
- **nuke.env ["numCPUs"]**. This returns the number of detected CPUs.
- **nuke.env ["gui"]**. This returns True if you are using the graphical user interface of Nuke, and False if you are running Nuke from a terminal or command prompt.
- **nuke.env ["ExecutablePath"]**. This returns the full path to Nuke's executable.
- **nuke.env ["ple"]**. This returns True if you are running the Personal Learning Edition of Nuke, and False if you are running the commercial version.
- **nuke.env ["WIN32"]**. This returns True if you are running Nuke on Windows, and False if you are not.
- **nuke.env ["MACOS"]**. This returns True if you are running Nuke on a Mac, and False if you are not.
- **nuke.env["LINUX"]**. This returns True if you are running Nuke on Linux, and False if you are not.
- **nuke.env ["64bit"]**. This returns True if you are running a 64-bit Nuke.

To find out what's in the Nuke "global environment", you can also use the following statement:
print nuke.env

Some of the items are read-only (for example, '64-bit' and 'ple'), and others can be set (for example, 'threads').

Accessing Node Metadata

To get a Python dictionary with all the metadata for a node (in this case, Read1), use:
nuke.toNode("Read1").metadata()

If you're interested in the metadata at a particular frame or in a particular view, you can also specify these as optional arguments:
nuke.toNode("Read1").metadata("key", frame, "view")

For example, say you have loaded a stereo pair of images in one Read node. If you want to find out the file modification time at frame 93 in the left view, you can do the following:

```
nuke.toNode("Read1").metadata("input/mtime", 93, "left")
```

Similarly, the following will get you the file size at frame 95 in the right view:

```
nuke.toNode("Read1").metadata("input/filesize", 95, "right")
```

To get the value for a particular key in the metadata, use:

```
nuke.toNode("Read1").metadata("key")
```

For example, `nuke.toNode("Read1").metadata("input/ctime")` will return the value for the key called input/ctime.

Creating Dialogs and Panels

You can create modal dialogs and non-modal panels. Modal dialogs pop up in their own window and will not let the user do anything outside the dialog until the dialog is closed. Non-modal panels float above everything else and can be docked into panes and saved with window layouts.

Dialogs and panels can be invoked at start-up or when the user selects a menu item that calls the panel.

Creating modal dialogs

To create a modal dialog with **OK** and **Cancel** buttons, read through the following instructions and have a look at the example in the end of the instructions on page 581.

To create a modal dialog:

1. Create a file called **menu.py** in your plug-in path directory if one doesn't already exist.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. In the menu.py file, write a Python class which extends the PythonPanel class. Use the following statements:

   ```
   class ClassName( nukescripts.PythonPanel ):
     def _init_( self ):
       nukescripts.PythonPanel._init_( self, "title", "ID" )
   ```

   where

   - *ClassName* is the name of your Python class.
   - *title* is the window title of your dialog.

- *ID* is a unique ID for the dialog. Using an ID when creating a modal dialog is optional. If you do use one, we recommend using a reverse domain name system, for example, uk.co.thefoundry.mydialog.

3. Use addKnob() to add knobs to your dialog. You can also use removeKnob() to remove a knob, knobs() to return a list of the knobs on the dialog, writeKnobs() to save knob values in a string, and readKnobs() to restore them from a string.

   Any knobs that can be added using the *Manage User Knobs* dialog can also be added to a dialog. This includes the following: Unsigned_Knob, Boolean_Knob, Int_Knob, ColorChip_Knob, String_Knob, EvalString_Knob, Multiline_Eval_String_Knob, File_Knob, BBox_Knob, Double_Knob, WH_Knob, Channel_Knob, Script_Knob, PyScript_Knob, ChannelMask_Knob, Enumeration_Knob, Tab_Knob, Font_Knob, Text_Knob, Axis_Knob, XYZ_Knob, Scale_Knob, XY_Knob, Format_Knob, LookupCurves_Knob, Keyer_Knob, Color_Knob, AColor_Knob, Eyedropper_Knob, Transform2d_Knob, Link_Knob, Box3_Knob, Help_Knob, Range_Knob, Pulldown_Knob, Bitmask_Knob, Histogram_Knob, and UV_Knob.

4. Write another function and use showModalDialog() to display the dialog. This adds **OK** and **Cancel** buttons to your dialog and makes the dialog modal.

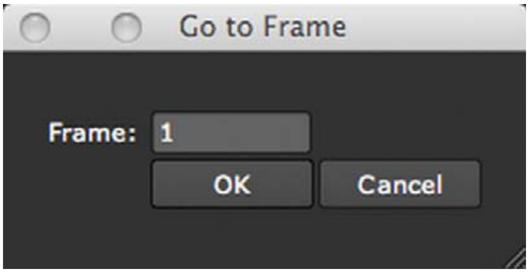
   You can also use the hide statement to hide the dialog.

5. Create a menu item that calls the previous function. For more information on how to do this, see "To define a menu or toolbar option:" on page 626.

   Alternatively, if you want to call the dialog at start-up, you can call the function after defining it.

Example

This example creates a modal dialog that has a **Frame** parameter and **OK** and **Cancel** buttons. The **Frame** parameter sets the frame to go to on the timeline. However, entering a new value for the **Frame** parameter does not change the current frame. The current frame is only changed when the user hits **OK**.



Frame: 1

Figure 20.4: The modal dialog created in this example.

Here's the example with comments:

```
import nukescripts
if nuke.env["gui"]:
```

# The following defines a new class called *ModalFramePanel*.

```
    class ModalFramePanel( nukescripts.PythonPanel ):
```

# The following function creates a dialog titled *Go to Frame* with the optional ID *uk.co.thefoundry.FramePanel.* It aso adds an integer knob called **frame** to the dialog. This knob is set to the value nuke.frame() which is the current frame on the timeline.

```
        def _init_( self ):
            nukescripts.PythonPanel._init_( self, "Go to Frame",
"uk.co.thefoundry.FramePanel" )
            self.frame = nuke.Int_Knob( "frame", "Frame:" )
            self.addKnob( self.frame )
            self.frame.setValue( nuke.frame() )
```

# The next function shows the dialog as a modal dialog. Doing this automatically adds the **OK** and **Cancel** buttons to the dialog.

```
        def showModalDialog( self ):
            result = nukescripts.PythonPanel.showModalDialog( self )
            if result:
                nuke.frame( self.frame.value() )
```

# The following function is called testModalPanel and tests whether the dialog works.

```
    def testModalPanel():
        return ModalFramePanel().showModalDialog()
```

# This last line calls the test function that then displays the dialog.

```
    testModalPanel()
```

# If you want to add the dialog to a menu item, you can also do the following:

```
menubar = nuke.menu("Nuke")
menubar.addCommand("&File/Show My Panel", testModalPanel)
```

Creating non-modal panels

To create a dockable panel that can be saved with window layouts, read through the following instructions and have a look at the example in the end of the instructions on page 584.

To create a non-modal panel:

1. Create a file called **menu.py** in your plug-in path directory if one doesn't already exist.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. In the menu.py file, write a Python class which extends the PythonPanel class. Use the following statements:

   ```
   class ClassName( nukescripts.PythonPanel ):
     def _init_( self ):
       nukescripts.PythonPanel._init_( self, "title", "ID" )
   ```

   where

   - **ClassName** is the name of your Python class.
   - **title** is the window title of your panel.
   - **ID** is a unique ID for the dialog used for saving the panel in window layouts. We recommend using a reverse domain name system to avoid clashes, for example, **uk.co.thefoundry.mypanel**.

3. Use **addKnob()** to add knobs to your panel. You can also use **removeKnob()** to remove a knob, **knobs()** to return a list of the knobs on the panel, **writeKnobs()** to save knob values in a string, and **readKnobs()** to restore them from a string.

   Any knobs that can be added using the *Manage User Knobs* dialog can also be added to a panel. This includes the following: Unsigned_Knob, Boolean_Knob, Int_Knob, Colorchip_Knob, String_Knob, EvalString_Knob, Multiline_Eval_String_Knob, File_Knob, BBox_Knob, Double_Knob, WH_Knob, Channel_Knob, Script_Knob, PyScript_Knob, ChannelMask_Knob, Enumeration_Knob, Tab_Knob, Font_Knob, Text_Knob, Axis_Knob, XYZ_Knob, Scale_Knob, XY_Knob, Format_Knob, LookupCurves_Knob, Keyer_Knob, Color_Knob, AColor_Knob, Eyedropper_Knob, Transform2d_Knob, Link_Knob, Box3_Knob, Help_Knob, Range_Knob, Pulldown_Knob, Bitmask_Knob, Histogram_Knob, and UV_Knob.

4. Write a second function that creates the panel when the user selects it from a content menu or restores a layout containing it. Use **addToPane()** to add the panel into the pane.

To get the pane, you should use **nuke.thisPane()**. This is only valid when executing a content menu command or when restoring a panel in a window layout.

To display the panel without adding it to the content menus and having the users select it from there, you can use the **show()** statement.

5. Create a menu item that calls the previous function. For example, you can add the menu item to the Pane menu. This way, your panel will be placed in the content menus where the panel will appear in a submenu called **Pane**.

   Alternatively, if you want to call the panel at start-up, you can call the function after defining it.

6. If you want your dialog to be saved with window layouts, you need to register your panel for that. In order for this to work at start-up, you must register your panel using a file called init.py because this file is read before the layout is restored. In your init.py file, enter **nukescripts.registerPanel (*ID, function*)**, where

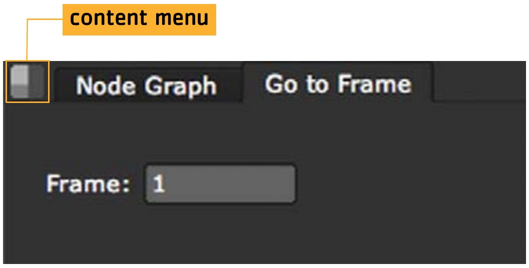
   - ***ID*** is the unique ID of the panel, for example **uk.co.thefoundry.mypanel**.
   - ***function*** is your Python function that creates the panel.

   Nuke will then save the ID in the layout file and try to call the function when the layout is restored.

   If you want to unregister your panel, you can use **nukescripts.unregisterPanel (ID, function)**.

Example

This example creates a panel that you can add to a pane by selecting **Pane** > **Frame Panel** from a content menu. The panel appears as a tab in that pane and can also be floated. The tab is called **Go to Frame**. It includes a **Frame:** parameter that allows the user to set the current frame on the timeline.



Frame: 1

Figure 20.5: The non-modal dialog created in this example.

Here’s the example with comments:

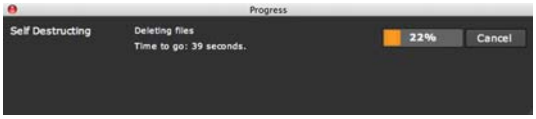
```
import nukescripts
# The following first defines a new class called FramePanel.
class FramePanel( nukescripts.PythonPanel ):
# The following is a function that creates the panel. The panel is given the
title Go to Frame and the ID uk.co.thefoundry.FramePanel. The ID will be
used for saving window layouts that contain this panel. An integer knob
called frame is also added to the panel. Its value is set to nuke.frame(),
which is the current frame on the timeline.
  def __init__( self ):
    nukescripts.PythonPanel.__init__( self, "Go to Frame",
"uk.co.thefoundry.FramePanel" )
    self.frame = nuke.Int_Knob( "frame", "Frame:" )
    self.addKnob( self.frame )
    self.frame.setValue( nuke.frame() )
# This function tells Nuke to change the current frame on the timeline if the
user changes the value of the frame knob.
  def knobChanged( self, knob ):
    if knob == self.frame:
      nuke.frame( self.frame.value() )
# The next function (called testPanel) will be called to create the panel when
the user selects it from a content menu or restores a layout containing it.
The function can either create a new panel or return a singleton if only one
such panel is allowed to exist.
def testPanel():
  return FramePanel().addToPane()
# Next, the testPanel function is added to the content menus where it will
appear under Pane > Frame Panel.
menu = nuke.menu( "Pane" )
menu.addCommand( "Frame Panel", testPanel )
# Finally, the panel's unique ID is registered for window layout restoration.
The testPanel function is called to create the panel that should go into the
pane with the unique ID.
nukescripts.registerPanel( "uk.co.thefoundry.FramePanel", testPanel )
```

Creating Progress Bar Dialogs

For lengthy tasks, you can create progress bar dialogs in the following manner:

```
import threading
import time
def selfDestruct():
  task = nuke.ProgressTask("Self Destructing")
  task.setMessage("Deleting files")
  for i in xrange( 0, 100 ):
    if task.isCancelled():
      nuke.executeInMainThread( nuke.message, args=( "Phew!" ) )
      break;
    task.setProgress(i)
    time.sleep( 0.5 )
threading.Thread( None, selfDestruct ).start()
```

This creates a progress bar dialog with the major message "Self Destructing" and the minor message "Deleting files". The isCancelled method returns True if the user has pressed the Cancel button.


Self Destructing
Deleting files
Time to go: 39 seconds.

There are a few important things to note here:

- You need to have the loop, because otherwise you'll never find out if the task was cancelled.
- You mustn't run this loop in Nuke's main thread (for example, by typing it into the Script Editor) because that will hang Nuke until the loop has finished.
- Therefore, you need to start a new thread to do your lengthy processing. This way, Nuke won't hang.
- However, you can't call nuke.message from a thread, so you need to use nuke.executeInMainThread() in order to display the message.

Clearing Out the Current Nuke (.nk) Script

To clear out the current Nuke script without opening a new Nuke window, use the following statement:

`nuke.scriptClear()`

Creating Views for a Stereoscopic Project

To create a new view for a stereoscopic or multi-view project, use the following statement:

`nuke.Root().addView("`***name***`")`

where ***name*** is the name of the new view. For example, to add a view called left, you should use the following: `nuke.Root().addView("left")`.

Adjusting Parameter Values in Stereo Projects

When you have created multiple views in your project, you can adjust node parameters separately for each view. To do so, you need to use the **setValue** statement, which takes a **view** argument.

For example, if you have created views called *left* and *right* in your project, you can do the following:

```
n = nuke.nodes.Blur()
n['size'].splitView()
n['size'].setValue(10, view = "left")
n['size'].setValue(20, view = "right")
```

This adds a Blur node into your script, splits the Size parameter off, and gives the left view the value 10 and the right view the value 20.

Automating Procedures

Okay, so you know how to use the Script Editor to type in a sequence of Python statements that take care of a procedure. But so far, you've still sat by your computer typing the statements in. It's clearly time to automate the procedure. All you need to do is save your statements, and when you want to use them again later, import them into the Script Editor.

To save statements in a Python module:

1. On the top of the Script Editor, click the **Save a script** button.
2. Save the script with the extension .py (for example **firstmodule.py**) in a directory that is contained in the sys.path variable. (To see these directories, enter **print sys.path** in the Script Editor. To add a directory to the sys.path variable, enter **sys.path.append (*"directory"*)** where ***directory*** represents the directory you want to add.)

You have now created your first Python module.

To open a Python script in the Script Editor:

1. Click the **Load a script** button. on top of the Script Editor. The **Script to open** dialog opens.
2. Navigate to the Python module that contains the script you want to open and click **Open**.

Nuke opens the script in the input pane of the Script Editor, but does not execute it.

To import and execute a Python script:

1. On top of the Script Editor, click the **Source a script** button The **Script to open** dialog opens.
2. Navigate to the Python module that contains the script you want to import and click **Open**.

OR

In the input pane, enter:
import *module*

where *module* represents the name of your Python module without the file extension, for example
import firstmodule

Nuke imports the Python module and performs the procedure defined in the module.

Note *Importing the module is done according to Python's default rules. During the import, the module is searched in the following locations and order:*

1. In the current directory.

2. In the directories contained in the ***PYTHONPATH*** *environment variable, if this has been defined. (To view these directories, enter* ***echo $PYTHONPATH*** *in a command shell.)*

3. In an installation-dependent default directory.

During the search, the variable ***sys.path*** *is initialized from these directories. Modules are then searched in the directories listed by the* ***sys.path*** *variable. To see these directories, execute the statement* ***print sys.path*** *in the Script Editor.*

Getting Help

In the scope of this user guide, it's not possible to go into detail with Python and all the scripts available. However, there are several sources of more information that you may find useful if you need help using Python.

More Documentation

To view documentation on Nuke's Python bindings, launch Nuke and select **Help** > **Documentation** > **Python Scripting**.

Viewing More Examples

We only described a few examples of Python scripts in this chapter, but there are more. You can find them in the following location:

- **On Windows**:
  drive letter:\Program Files\Nuke6.2v2\plugins\nukescripts or
  drive letter:\Program Files (x86)\Nuke6.2v2\plugins\nukescripts
- **On Mac OS X**:
  /Applications/Nuke6.2v2/Nuke6.2v2.app/Contents/MacOS/plugins/nukescripts
- **On Linux**:
  /usr/local/Nuke6.2v2/plugins/nukescripts

To view an example, select one of the .py files and open it in any text editor.

Using the Help Statement

Possibly the quickest way of getting help on the available scripts is to enter the following in the Script Editor:
help (nuke)

This statement lists many of the available Python commands in an alphabetical order together with their descriptions.

You can also get help on more specific things. For example, the statement
help (nuke.Knob.setValue)

would give you a description of what the **setValue (...)** method does.

Python on the Web

To read more about Python, check out its documentation, or interact with other Python users, visit the Python programming language official web site at http://www.python.org/.

Python Callbacks Embedded in Scripts and Nodes

Using the nuke.add...() functions described below, you can have Python functions automatically called when various events (such as creating a node or loading a script) happen in Nuke.

You can use all the nuke.add...() functions in your init.py or menu.py file. This way, the code is considered part of your Nuke environment and does not vary from script to script.

All the nuke.add...() functions take the same arguments, for example:
nuke.addOnCreate (*callable*, *args*=( ), *kwargs*={}, *nodeClass*='*')

where

- ***callable*** is a Python callable, such as the name of a defined function.
- ***args*** is the list of arguments. These should be in parenthesis. For example, **nuke.addOnCreate(nuke.tprint, ('what to print'))**.
- ***kwargs*** is a list of arguments given as keyword+value pairs. For example, **nuke.addOnCreate(nuke.tprint, ('a', 'b'), { 'sep': ',' })**.
- ***nodeClass***. The code will only be called if nuke.thisNode().nodeClass() equals this string. For example, **nuke.addOnCreate(nuke.tprint, ('Creating Write'), nodeClass='Write')**. The default value of '*' means that this is called for all nodes.

During the callback, there is a context node (the node to which the action is happening to), which can be examined using nuke.thisNode(). For knobChanged, there is also a context knob, which you can get using nuke.thisKnob().

For many of the nuke.add...() functions, such as onCreate, there are also knobs with the same name. These knobs fall into two categories:

- Visible knobs. These are the knobs that the artists are expected to edit. They can be seen on the **Python** tab of some properties panels. The visible knobs include onScriptLoad, onScriptSave, onScriptClose, beforeRender, beforeFrameRender, afterRender, and afterFrameRender.
- Hidden knobs. These are not visible in any properties panels, but can be set using Python. These are intended to allow programming the behavior of gizmos. Artists should not set these knobs, as the user settings will override any saved gizmo settings. The hidden knobs include onCreate, onDestroy, knobChanged, updateUI, and autolabel.

If there are many callbacks registered, code attached to knobs is always called first before the nuke.add...() functions. (For example, code put in the onCreate knob is called before the nuke.addOnCreate() function.) In most cases, it is then followed by all the nuke.add...() functions with a matching nodeClass in the order they were added, and finally the ‘*’ ones also in the order they were added. Note that the addAutolabel() and addFilenameFilter() functions are exceptions to this rule, however, as they are called in reverse order.

All the add callbacks calls have a corresponding remove callback call.

The currently supported nuke.add...() functions and knobs are described below. The corresponding nuke.remove...() functions are also listed.

onUserCreate

nuke.addOnUserCreate(function), nuke.removeOnUserCreate(function)

This is executed whenever a node is created *by the user using the Nuke graphical user interface (GUI)*. It is also called at start-up on the Root and Viewer nodes. It is not called when loading existing scripts, pasting nodes, or undoing a delete.

You can use this code to change the default values of knobs, to add user knobs, and to perform other actions to preset nodes to the way you want users to see them.

Nuke.addOnUserCreate is run immediately after the knobs are set to their default values (including knobDefault values). It is run before onCreate. If nuke.addOnUserCreate() is not used, nuke.tcl("OnCreate") is called for back-compatibility.

onCreate

node.knob('onCreate'), nuke.addOnCreate(function), nuke.removeOnCreate(function)

This is executed when *any* node is created, for example, when loading a script, pasting a node, selecting a menu item, or undoing a delete.

Note that if an onCreate function is defined before a script is loaded, it will be run for each node as the nodes are created while the script is loaded.

When a group (including the root) is created with a number of inner nodes, onCreate is called first on the inner nodes, then on the group itself. For root, this is run when any script is loaded (see onScriptLoad below) or when **File** > **New** is selected. For the preferences and Python knob panels, this is run when they are created.

Example

The following code will make the creation of every node print *OnCreate called for [node name]* on the terminal:

```
nuke.addOnCreate(lambda: nuke.tprint("OnCreate called for "+nuke.thisNode().name()))
```

onScriptLoad

root.knob('onScriptLoad'), nuke.addOnScriptLoad(function), nuke.removeOnScriptLoad(function)

This is executed when any script is loaded. It is run immediately after the onCreate for the root. It is not run for "new" scripts.

The onScriptLoad knob is visible on the **Python** tab of the Project settings.

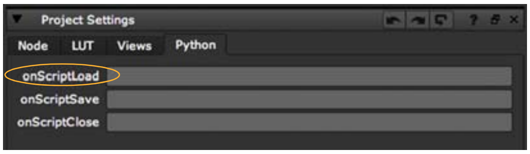

Project Settings
Node
LUT
Views
Python
onScriptLoad
onScriptSave
onScriptClose

Figure 20.6: The Python tab in the Project settings.

onScriptSave

root.knob('onScriptSave'), nuke.addOnScriptSave(function), nuke.removeOnScriptSave(function)

These are run when the user tries to save a script. If this changes the script name, that is the name the script is saved under.

The onScriptSave knob is visible on the **Python** tab of the Project settings.

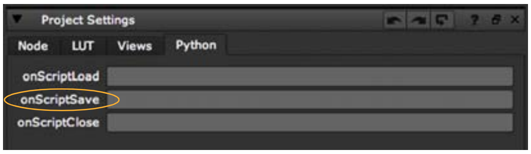

Project Settings
Node
LUT
Views
Python
onScriptLoad
onScriptSave
onScriptClose

Figure 20.7: The Python tab in the Project settings.

onScriptClose

root.knob('onScriptClose'), nuke.addOnScriptClose(function), nuke.removeOnScriptClose(function)

These are run when the user exits Nuke or closes a script. They are also run by the scriptClear() function. They are run immediately before the onDestroy for the root.

The onScriptClose knob is visible on the **Python** tab of the Project settings.

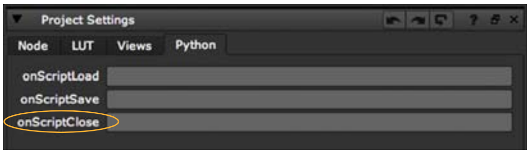

Project Settings
Node
LUT
Views
Python
onScriptLoad
onScriptSave
onScriptClose

Figure 20.8: The Python tab in the Project settings.

onDestroy

node.knob('onDestroy'), nuke.addOnDestroy(function), nuke.removeOnDestroy(function)

These are executed when any node is deleted, including when the creation of a node is undone. They are called on the root when a script is closed, or when the user exits Nuke. They are NOT run for the Preferences or Python knob panels, or if Nuke crashes.

For groups and the root, this is run before all the children.

knobChanged

node.knob('knobChanged'), nuke.addKnobChanged(function), nuke.removeKnobChanged(function)

These are executed when the user changes the value of any knob and *the control panel is open*. They are not called recursively. You can find the knob that was changed by using nuke.thisKnob().

The knobChanged knob allows you to make gizmos with knobs that have effects like enabling, disabling, or presetting other knobs.

To access the knob that was changed, use nuke.thisKnob().

Note *The purpose of knobChanged is to update the control panel appearance. You cannot use it to "track" knobs (for example, to keep a database up-to-date with the values of all the knobs), as it is not called when the properties panel is closed. To track knobs, use updateUI (described below) instead.*

When the user opens or closes the properties panel, or when they change the input connections while the panel is open, you can call knobChanged with the "dummy" knobs named "showPanel" and "inputChange". If you are disabling or otherwise altering values of knobs on changes, you probably need to respond to showPanel to get the panel into the correct state for the current settings when it is opened. For example, you could do:

```
# if slider is zero, disable the checkmark:
n = nuke.thisNode()
k = nuke.thisKnob()
if k.name()=="slider" or k.name()=="showPanel":
  n['checkmark'].setEnable(n['slider'].getValue()!=0)
```

Example

You can use nuke.addKnobChanged to gang the gain and gamma sliders of the selected ColorCorrect node so that if the user adjusts one slider, the other is automatically set to the same value. To gang the sliders, use the following code:

```
def gangGammaGainSliders():
 n = nuke.thisNode()
 k = nuke.thisKnob()
 if k.name() == "gamma":
   n['gain'].setValue(k.value())
 elif k.name() == "gain":
   n['gamma'].setValue(k.value())
nuke.addKnobChanged(gangGammaGainSliders, nodeClass="ColorCorrect")
```

updateUI

node.knob('updateUI'), nuke.addUpdateUI(function), nuke.removeUpdateUI(function)

These are run on every node after any changes to the script. This is done as a low-priority process during idle and thus Nuke may have already started calculating the image for the Viewer before updateUI is called. Therefore, updateUI should not make the kind of changes to the script that will change the image (otherwise, the Viewer would have to restart).

This is run just before autolabel and after knobChanged. To avoid running this on every node, Nuke examines what updateUI looks at (such as the frame or view number) to decide whether or not to run it. If the function does not refer to the frame or view number, Nuke will only call it when the knob values on the node are changed, and not if the frame or view changes.

autolabel

node.knob('autolabel'), nuke.addAutolabel(function), nuke.removeAutolabel(function)

These are run immediately after updateUI and return the string to draw on the node in the Node Graph. If the autolabel knob is not blank and returns anything other than None, the return value is displayed on the node. Otherwise, the nuke.addAutolabel() functions are called in reverse order and the first one to return anything other than None is used. If all of them return None, then nuke.thisNode().name() will be used.

For back-compatibility, Nuke does addAutolabel(__main__.autolabel) on start-up.

beforeRender

write.knob('beforeRender'), nuke.addBeforeRender(function), nuke.removeBeforeRender(function)

These are run prior to starting rendering in execute(). If they throw an exception, the render aborts. They can be used to create the destination directory or communicate with your render farm, for example.

The beforeRender knob (**before render**) is visible on the **Render** tab of Write nodes.

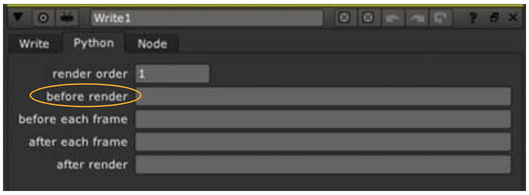
Tip *By default, Nuke doesn't create directories when rendering files. If you have a Write node whose* ***file*** *control points to a directory that doesn't exist, the render will fail. However, you can change this behavior by using the function* ***nuke.addBeforeRender()*** *to register a function that creates the necessary directory prior to rendering the first frame. Here's an example of how to do so:*

1. Create a file called init.py in your plug-in path directory if one doesn't already exist.

For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. Open the init.py file in a text editor and add an entry in the following format:

```
def createWriteDir():
  import nuke, os
  file = nuke.filename(nuke.thisNode())
  dir = os.path.dirname( file )
  osdir = nuke.callbacks.filenameFilter( dir )
  os.makedirs( osdir )
nuke.addBeforeRender(createWriteDir)
```



render order 1

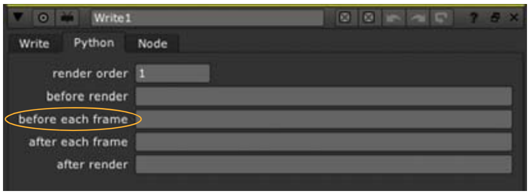
Figure 20.9: The Python tab in the Write node properties.

beforeFrameRender

write.knob('beforeFrameRender'), nuke.addBeforeFrameRender(function), nuke.removeBeforeFrameRender(function)

These are run prior to starting rendering of each individual frame. If they throw an exception, the render aborts. They can be used to create the destination directory or communicate with your render farm, for example.

The beforeFrameRender knob (**before each frame**) is visible on the **Render** tab of Write nodes.


Write1
Write
Python
Node
before render
before each frame
after each frame
after render

render order 1

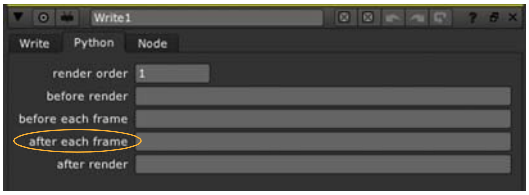
Figure 20.10: The Python tab in the Write node properties.

afterFrameRender

write.knob('afterFrameRender'), nuke.addAfterFrameRender(function), nuke.removeAfterFrameRender(function)

These are run after *each frame* is finished rendering. They are not called if the render aborts. If they throw an exception, the render aborts. They can be used to copy each frame to a digital video recorder (DVR), for example.

The afterFrameRender knob (**after each frame**) is visible on the **Render** tab of Write nodes.


Write1
Write
Python
Node
before render
before each frame
after each frame
after render

render order 1

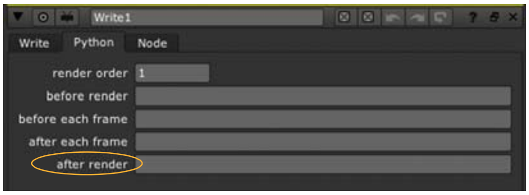
Figure 20.11: The Python tab in the Write node properties.

afterRender

write.knob('afterRender'), nuke.addAfterRender(function), nuke.removeAfterRender(function)

These are run after rendering of *all frames* is finished. If they throw an error, the render aborts. They are useful for finishing up a copy to a digital video recorder (DVR) or for communicating with a render farm, for example.

The afterRender knob (**after render**) is visible on the **Render** tab of Write nodes.



render order 1

Figure 20.12: The Python tab in the Write node properties.

filenameFilter

nuke.addFilenameFilter(function), nuke.removeFilenameFilter(function)

This adds a filter function that processes any file names that are in the Nuke script before passing the script to the system. These filters can be used to remove differences between operating systems or to insert required portions of path names. This callback must take a single string argument, and can either return a new string or None (which is the same as returning the string unchanged). For filenames that are not for a specific node, such as plug-in names, this function is called with nuke.thisNode() set to the root node. All the functions passed to nuke.addFilenameFilter are called in reverse order (the one added last is called first).

For back-compatibility, if no functions have been added, Nuke runs main.filenameFix(s). The default version of this function in turn calls nuke.tcl('filename_fix',s).

Using wxPython in Nuke to Write Custom GUIs

WxPython is a collection of Python bindings for the graphical user interface (GUI) toolkit wxWidgets. Currently, Nuke works with wxPython 2.8 on Windows, Linux, and Mac.

You can use wxPython to write custom wxWidgets applications or GUIs in Python and use them within Nuke. WxPython needs to be installed separately to Nuke. You can get a copy of wxPython from http://wxpython.org/. WxPython includes wxWidgets, so you do not need to install the latter.

Both wxPython and wxWidgets are open-source software that you can download and use free of charge.

Installation

To use wxPython in Nuke, you need to install the following tools.

On Windows

1. Install Python 2.6.5 from www.python.org.
2. Install wxPython for 2.6 unicode from http://wxpython.org/. You can install it anywhere you like as long as you point the sys.path variable to it, as described in step 3. Only if you install wxPython inside *drive letter*:\Program Files\Nuke6.2v2\lib\site-packages or *drive letter*:\Program Files (x86)\Nuke6.2v2\lib\site-packages, you do not need to perform step 3.
3. Point the sys.path variable to where the wxPython modules are installed. To see the directories contained in the sys.path variable, enter **print sys.path** in the Script Editor. To add a directory to the sys.path variable, enter **sys.path.append ("directory")**. In this statement, **directory** represents the directory you want to add, for example **sys.path.append ("C:\\Python26\\Lib\\site-packages\\wx-2.8-msw-unicode")**.

On Mac OS X

1. Install Python 2.6.5 from www.python.org.
2. Install wxPython for 2.6 unicode from http://wxpython.org/.

   You can install Python and wxPython anywhere you like as long as you point the sys.path variable to them, as described in step 3.
3. Point the sys.path variable to where the Python and wxPython modules are installed. To see the directories contained in the sys.path variable, enter **print sys.path** in the Script Editor. To add a directory to the sys.path variable, enter **sys.path.append ("directory")**. In this statement, **directory** represents the directory you want to add, for example
   - **sys.path.append ("/System/Library/Frameworks/Python.framework/Versions/2.6/Extras/lib/python")**
   - **sys.path.append ("/System/Library/Frameworks/Python.framework/Versions/2.6/Extras/lib/python/wx-2.8-mac-unicode")**

On Linux

1. Install Python 2.6.5 from www.python.org.
2. Install wxPython for 2.6 unicode from http://wxpython.org/.

   You can install Python and wxPython anywhere you like as long as you point the sys.path variable to them, as described in step 3. Only if you install them inside /usr/local/Nuke6.2v2/site-packages, you do not need to perform step 3.

3. Point the sys.path variable to where the Python and wxPython modules are installed. To see the directories contained in the sys.path variable, enter **print sys.path** in the Script Editor. To add a directory to the sys.path variable, enter **sys.path.append ("directory")**. In this statement, **directory** represents the directory you want to add, for example
   - **sys.path.append ("/usr/lib/python2.6/site-packages")**
   - **sys.path.append ("/usr/lib/python2.6/site-packages/wx-2.8-gtk2-unicode")**

General Workflow

After you've installed the necessary tools, you are ready to use wxPython in Nuke. Do the following:

1. Write an application using wxPython. Save the application in a .py file in a directory that's contained in the sys.path variable, for example in $HOME/.nuke. (To see these directories, enter **print sys.path** in the Script Editor. To add a directory to the sys.path variable, enter **sys.path.append ("directory")** where directory represents the directory you want to add.)
2. To be able to use a helper script that you'll need in the next step, enter the following in your .py file:

   from nukescripts import utils, pyWxAppUtils
3. Pass the application to the pyWxAppUtils.py helper script in the following manner:

   pyWxApp = pyWxAppUtils.pyWxAppHelper(MyApp, start = True)

   The pyWxAppUtils.py is a helper script that executes a wxPython application from a separate thread in Nuke. You can find the script in the following location:
   - **On Windows**:

     drive letter:\Program Files\Nuke6.2v2\plugins\nukescripts or

     drive letter:\Program Files (x86)\Nuke6.2v2\plugins\nukescripts
   - **On Mac OS X**:

     /Applications/Nuke6.2v2/Nuke6.2v2.app/Contents/MacOS/plugins/nukescripts
   - **On Linux**:

     /usr/local/Nuke6.2v2/plugins/nukescripts.
4. In order to communicate with Nuke's event loop, make sure any Nuke Python calls go through either **nukescripts.utils.executeInMainThread** or **nukescripts.utils.executeInMainThreadWithResult**. For example, to create a Blur node, you could use:

- **nukescripts.utils.executeInMainThread(nuke.nodes.Blur)**

  or if the result of the call is needed (for example, in this case to get the node object):
- **b = nukescripts.utils.executeInMainThreadWithResult(nuke.nodes.Blur)**

Note *You should only use these calls when interacting with Nuke from a separate thread. Otherwise, Nuke may hang.*

Using PyQt4 in Nuke to Write Custom GUIs

PyQt is a collection of Python bindings for Qt Development Frameworks's graphical user interface (GUI) toolkit Qt. PyQt is developed by Riverbank Computing, and combines the advantages of Python and Qt. Currently, Nuke works with PyQt4 on Windows and Linux but not on Mac OS X.

With PyQt, you can write custom Qt applications or GUIs in Python and use them within Nuke. Because many other applications, such as Maya and Houdini, also support PyQt, you can share the same UI across these applications without having to replicate or build it separately for each of them.

Both PyQt and Qt need to be installed separately to Nuke. You can get a copy of PyQt from www.riverbankcomputing.co.uk, and a copy of Qt from http://qt.nokia.com/.

Installation

To install the following tools, follow the instructions given by Qt Software and Riverbank Computing.

1. Obtain a copy of Qt from http://qt.nokia.com/ and install it on your machine. Qt is a C++ GUI library.
2. Download and install SIP from www.riverbankcomputing.co.uk. SIP is a tool for creating Python bindings for C++ libraries. Make sure you compile it against Python 2.6.5.
3. Download and install a copy of PyQt4 from www.riverbankcomputing.co.uk. PyQt is a collection of Python bindings for Qt.

Where you install these applications does not matter as long as the sys.path variable points to where the Python modules are installed. (To see the directories contained in the sys.path variable, enter **print sys.path** in the Script Editor. To add a directory to the sys.path variable, enter **sys.path.append ("directory")**, where **directory** represents the directory you want to add.)

General Workflow

This is the general workflow for using PyQt in Nuke:

1. Initialize PyQt from a separate thread in Nuke. An example of how to do this can be found in a file called pyQtAppUtils.py in the nukescripts folder. See "PyQtAppUtils.py: Launching PyQt from a separate thread in Nuke" on page 602.
2. Start a separate GUI application (that is, any application that you have written to control Nuke features) and have it run concurrently with Nuke. An example of how to do this can be found in a file called pyQtRender.py in the nukescripts folder. For more information, see "PyQtRender.py: Using a GUI built in QtDesigner" on page 602.

   To be able to use the pyQtAppUtils.py example script that initializes PyQt from a separate thread, make sure you use the following from the pyQtRender.py file in your own projects:

   from nukescripts import utils, pyQtAppUtils
3. In order to communicate with Nuke's event loop, make sure any Nuke Python calls go through either **nukescripts.utils.executeInMainThread** or **nukescripts.utils.executeInMainThreadWithResult**. For example, to create a Blur node, you could use:
   - **nukescripts.utils.executeInMainThread(nuke.nodes.Blur)**

     or if the result of the call is needed (for example, in this case to get the node object):
   - **b = nukescripts.utils.executeInMainThreadWithResult(nuke.nodes.Blur)**

Note *The executeInMainThread calls take arbitrary Python and run that in Nuke's main thread.*

They're not supposed to be called per every Nuke Python call made. They can and should be batched. Otherwise, they may have unpredictable effects as Nuke is running Python and may acquire the lock and interrupt.

Example

In the nukescripts folder, there are example modules that illustrate how to use PyQt to create a custom GUI inside Nuke. You can find the modules in the following location:

- **On Windows**:

  drive letter:\Program Files\Nuke6.2v2\plugins\nukescripts or

  drive letter:\Program Files (x86)\Nuke6.2v2\plugins\nukescripts
- **On Linux**:

  /usr/local/Nuke6.2v2/plugins/nukescripts

The example consists of two modules: pyQtAppUtils.py and pyQtRender.py. PyQtAppUtils.py launches PyQt from a separate thread in Nuke, while pyQtRender.py is an example of how to use a GUI built in QtDesigner. You can run both modules by entering the following in the Script Editor:

```
from nukescripts.pyQtExamples import pyQtRender
pyQtRender.startQtRenderDialog()
```

What the two modules do is described below in more detail.

PyQtAppUtils.py: Launching PyQt from a separate thread in Nuke

The pyQtAppUtils.py file is an example of a module that initializes PyQt from a separate thread in Nuke. To see what the module contains, open the pyQtAppUtils.py file from the nukescripts folder in a text editor.

PyQtRender.py: Using a GUI built in QtDesigner

The pyQtRender.py module first imports the pyQtAppUtils.py module that launches PyQt from a separate thread in Nuke. It then creates a dialog with a Render button. The Render button renders the Write node that is currently selected in the dialog.

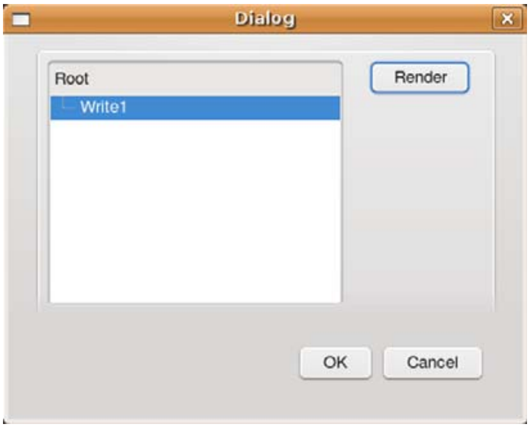


Figure 20.13: The pyQtRender.py example uses a dialog built in QtDesigner.

To see what this module contains, open the pyQtRender.py file from the nukescripts folder in a text editor.

21 CONFIGURING NUKE

This chapter shows visual effects supervisors how to configure Nuke for multiple artists, prior to the start of a project. These are the common application settings discussed in this chapter:

- Command line operations
- Environment variables
- Gizmo, NDK plug-in, and TCL script directories
- Python script directories
- OFX plug-in directories6.2v2
- Favorite directories
- Cross-platform file paths
- Menu and Toolbar options
- Image formats
- Gizmos (Nuke group nodes or subscripts that allow only select modifications)
- Custom plug-ins (binary plug-ins made via the Nuke software developers kit)
- Generic TCL ("TiCkLe") scripts
- Template scripts
- Common preferences
- Script's lookup tables (LUTs)
- Custom Viewer Processes

Note *If you copy and paste Python example scripts from this user guide into a text editor, line indentations may not be preserved. If this is the case, correct the indentations manually.*

What Is a Terminal and How Do I Use One?

Many tasks in this chapter tell you to enter commands from a terminal or shell. This refers to a window where you can enter commands directly rather than making selections through a user interface.

The following describes how to open such a window for your operating system.

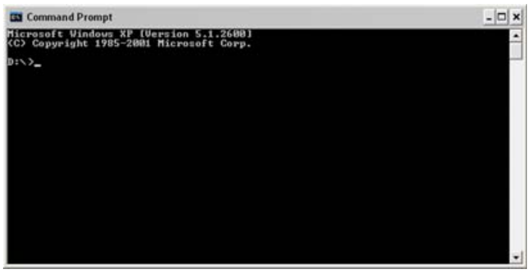
- **Linux**: Click the right mouse button over the desktop and choose **New Terminal** (or **Open Terminal**) from the pop-up menu.
- **Windows**: From the **Start** menu, choose **All Programs > Accessories > Command Prompt**.

- Mac OS X: Click on the **Terminal** dock icon.

  OR

  Browse to the **Applications > Utilities** folder on your system hard drive, and double-click the **Terminal** icon.

Inside the terminal or shell, you'll see a command prompt, which looks similar to this:



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

D:\>_
```

Figure 21.1: Command prompt window for Windows XP.

Once you see the command prompt, you can enter commands to perform various tasks like listing the files in a directory or running programs. Here are some specific examples:

- On Linux or Mac OS X, type **pwd** and press **Enter** to view the path of the current directory. On Windows, the equivalent command would be **cd**.
- On Linux or Mac OS X, type **`ls`** and press **Enter** to view a list of files in the current directory. On Windows, the equivalent command would be **`dir`**.
- On Linux, Mac OS X and Windows, type **cd** followed by a full pathname and press **Enter** to change directories.

Command Line Operations

Command-line flags activate various options when you launch Nuke from a shell, and provide additional functionality to Nuke. First let's discuss how to launch Nuke from a shell.

On Mac OS X

Open a Terminal and change directory as follows:

```
cd /Applications/Nuke6.2v2/Nuke6.2v2.app/
```

To launch Nuke, type this command:

```
./Nuke6.2v2
```

Alternatively, you can set an alias to point to Nuke and then you can launch

Nuke from any directory. The procedure for this depends on what your default shell is. To get the name of the shell you are using, launch Terminal and enter **`echo $SHELL`**.

If you are using a tcsh shell, enter:

```
alias nuke /Applications/Nuke6.2v2/Nuke6.2v2.app/Nuke6.2v2
```

(On 32-bit Nuke)

```
alias nuke /Applications/Nuke6.2v2/Nuke6.2v2.app/Nuke6.2v2
```

(On 64-bit Nuke)

Alternatively, if you are using a bash shell enter:

```
alias nuke='/Applications/Nuke6.2v2-32/Nuke6.2v2.app/
Nuke6.2v2'
```

(On 32-bit Nuke)

```
alias nuke='/Applications/Nuke6.2v2/Nuke6.2v2.app/
Nuke6.2v2'
```

(On 64-bit Nuke)

If you want to launch NukeX, enter:

```
alias nukex /Applications/Nuke6.2v2-32/NukeX6.2v2.app/
NukeX6.2v2
```

(On 32-bit Nuke)

```
alias nukex /Applications/Nuke6.2v2/NukeX6.2v2.app/
NukeX6.2v2
```

(On 64-bit Nuke)

Change to your HOME directory:

```
cd
```

Launch nuke:

```
nuke
```

Tip *You can add aliases to a .cshrc file (if you're using a tcsh shell) in your home directory so that they are activated each time you open a shell. See your Systems Administrator for help setting this up.*

If you're on Mac OS X 10.5 or later, then your default shell is BASH and the file you need to edit is the .bash_profile file to add an alias to your home directory. Edit your .bash_profile to include this line:

```
alias nuke="/Applications/<your home directory>"
```

Note however that .bash_profile isn't read by bash in all cases. It's only read if bash is invoked as a login shell. Therefore it may be necessary to also add the alias to .bashrc.

To prevent two alias lists being kept in your system, you can create a file called .aliasrc with this line:

```
alias nuke="/Applications/<your nuke directory>"
```

Then in both .bash_profile and .bashrc include the following line:

```
. .aliasrc
```

Now you can start experimenting with command line flags on launching Nuke. Here's one that displays the version number and build date.

```
nuke -version
```

If you have a Nuke script, you can render it on the command line without opening the GUI version. Here's an example that renders a hundred frames of a Nuke script:

```
nuke -F 1-100 -x myscript.nk
```

Note how you can use the -F switch on the command line to indicate a frame range, and separate the starting and ending frames with a dash.

Note *We recommend that you use the new -F switch whenever defining a frame range on the command line. However, for backwards compatibility, you can also use the old syntax in this release. To do so, place the frame range in the end of the command and use a comma to separate the starting and ending frames. For example:*

```
nuke -x myscript.nk 1,100
```

For more information on defining frame ranges, see "Defining Frame Ranges" on page 140.

To display a list of command line flags (switches) available to you, use the following command:

```
nuke -help
```

Here's that list of command line flags in a table:

Switch/Flag	Action
-b	Background mode. This launches Nuke and returns control to the terminal, so you get your prompt back. This is equivalent to appending a command with an & to run in the background.
-c *size (***k***,* **M***, or* **G***)*	Limit the cache memory usage, where *size* equals a number in bytes. You can specify a different unit by appending **k** (kilobytes), **M** (megabytes), or **G** (gigabytes) after *size*.
-d <x server *name*>	This allows Nuke to be viewed on one machine while run on another. (Linux only and requires some setting up to allow remote access to the X Server on the target machine).

Switch/Flag	Action
`-f`	Open Nuke script at full resolution. Scripts that have been saved displaying proxy images can be opened to show the full resolution image using this flag. See also -p.
`-F`	Frame numbers to execute the script for. Here are some examples: • -F 3 indicates frame 3. • -F 1-10 indicates frames 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10. • -F 1-10x2 indicates frames 1, 3, 5, 7, and 9. You can also use multiple frame ranges, such as -F 1-50x1 -F 51-60x2 -F 60-100x3.
`-h`	Display command line help.
`-help`	Display command line help.
`-i`	Use an interactive (nuke_i) FLEXlm license key. This flag is used in conjunction with background rendering scripts using -x. By default -x will use a nuke_r license key, but -ix will background render using a nuke_i license key.
`-l`	New read or write nodes will have the colorspace set to linear rather than default.
`-m` *#*	Set the number of threads to the value specified by *#*.
`-n`	Open script without postage stamps on nodes.
`-p`	Open Nuke script at proxy resolution. Scripts that have been saved displaying full resolution images can be opened to show the proxy resolution image using this flag. See also -f.
`-q`	Quiet mode. This stops all printing to the shell.
`-s` *#*	Sets the minimum stack size, or the node tree stach cache size for each thread in bytes. This defaults to 16777216 (16 MB). The smallest allowed value is 1048576 (1 MB).
`-t`	Terminal mode. This allows you to enter Python commands without launching the GUI. A >>> command prompt is displayed during this mode. Enter quit() to exit this mode and return to the shell prompt. This mode uses a nuke_r license key by default, but you can get it to use a nuke_i key by using the -ti flag combo.
`-V`	Verbose mode. In the terminal, you'll see explicit commands as each action is performed in Nuke.
`-v`	This command displays an image file inside a Nuke Viewer. Here's an example: nuke -v image.tif
`-version`	Display the version information in the shell.

Switch/Flag	Action
-x	eXecute mode. Takes a Nuke script and renders all active Write nodes. Note that it is not possible to render a PLE (Personal Learning Edition) script with -x from the command line. Note also that this mode uses a FLEXlm nuke_r license key. To use a nuke_i license key, use -xi. This is the syntax: nuke -x myscript.nk On Windows, you can press **Ctrl+Break** to cancel a render without exiting if a render is active, or exit if not. **Ctrl/Cmd+C** exits immediately. On Mac and Linux, **Ctrl/Cmd+C** always exits.
-X *node*	Render only the Write node specified by *node.*
--	End switches, allowing script to start with a dash or be just - to read from stdin

General syntax

This is the general syntax for using these options when launching Nuke at the command prompt:

nuke *<switches> <script> <argv> <ranges>*

<switches> - modifies the behavior of Nuke when run from the command line. A list of switches is given in the table above. These are sometimes called flags.

<script> - the name of the Nuke script.

<argv> - an optional argument that can be used in Nuke. See the example below.

<ranges> - this is the frame range you want rendering.

Examples

Let's consider some practical examples.

To launch Nuke and open a script.

```
nuke myscript.nk
```

Crazy I know, but I've called my script, -myscript.nk, and the hyphen at the start of the filename has confused Nuke. To get round this if you don't want to rename your file use the double hyphen syntax:

```
nuke -- -myscript.nk
```

Viewing images

To display an image:

```
nuke -v polarbear.tif
```

To display several images:

```
nuke -v polarbear.tif whiteflower.psd mountains.cin
```

To display an image sequence (taxi.0001.tif, taxi.0002.tif,...,taxi.0050.tif):

```
nuke -v taxi.####.tif 1-50
```

Rendering on the command line

To render frame 5 of a Nuke script:

```
nuke -F 5 -x myscript.nk
```

To render frames 30 to 50 of a Nuke script:

```
nuke -F 30-50 -x myscript.nk
```

To render every tenth frame of a 50 frame sequence of a Nuke script:

```
nuke -F 1-50x10 -x myscript.nk
```

This will render frames 1, 11, 21, 31, 41.

In a script with two write nodes called WriteBlur and WriteInvert this command will just render frames 1 to 20 from the WriteBlur node:

```
nuke -X WriteBlur myscript.nk 1-20
```

Using [argv 0]

Let's use [argv] to vary the output file name. Launch the GUI version of Nuke and create a node tree that puts a checker into a Write node. Open the write node property panel by double clicking on it and in the file text field enter this filename:

[argv 0].####.tif

Save the script and quit Nuke. On the command line type:

```
nuke -x myscript.nk mychecker 1-5
```

This will render 5 frames (mychecker.0001.tif, mychecker.0002.tif, etc.).

You can add another variable to control the output image file type. The file text field needs this:

[argv 0].####.[argv 1]

and then render the script using this command:

```
nuke -x myscript.nk mychecker cin 1-5
```

to get mychecker.0001.cin, mychecker.0002.cin, etc.

The <argv> string can be any [argv n] expression to provide variable arguments to the script. These must be placed between the *<script>* and the *<ranges>* on the command line. You can include multiple expressions, but each must begin with a non-numeric character to avoid confusion with the frame range control. For more information on expressions, see Chapter 18: *Expressions* on page 533.

Using Python to convert TIFFs to JPGs

This command line method will convert 5 tiff frames to jpeg.

```
nuke -t
>>> r = nuke.nodes.Read(file = ”myimage.####.tif”)
>>> w = nuke.nodes.Write(file = ”myimage.####.jpg”)
>>> w.setInput( 0, r )
>>> nuke.execute(“Write1”, 1,5)
>>> quit()
```

It’s a bit tedious typing these commands in line by line. So let’s put them in a text file called imageconvert.py and get Nuke to execute the Python script.

```
cat imageconvert.py
r = nuke.nodes.Read(file = ”myimage.####.tif”)
w = nuke.nodes.Write(file = ”myimage.####.jpg”)
w.setInput( 0, r )
nuke.execute(“Write1”, 1,5)

nuke -t < imageconvert.py
```

You can also pass in the Python script as a command line parameter. Doing this will allow you to enter additional parameters after the script name to pass into your script. When you do so, note that sys.argv[0] will be the name of the Python script being executed, and argv[1:] will be the other parameters you passed in. One example of this is below. See the standard Python module optparse for other ways to parse parameters.

```
cat imageconvertwithargs.py

import sys
r = nuke.nodes.Read(file = sys.argv[1])
w = nuke.nodes.Write(file = sys.argv[2])
w.setInput(0, r)
nuke.execute("Write1", 1, 5)

nuke -t imageconvertwithargs.py myimage.####.tif
myimage.####.jpg
```

Environment Variables

Environment variables are named variables used to store a value, such as a specific file path. They can be used to influence Nuke's behavior. For example, Nuke uses the information stored in them to define where to place certain files.

Setting Environment Variables

To set an environment variable:

On Windows

1. Right-click on **My Computer** and select **Properties**.
2. Go to the **Advanced** tab.
3. Click the **Environment Variables** button. The *Environment Variables* dialog opens.
4. Click the **New** button under either **User variables** or **System variables**, depending on whether you want to set the variable for the current user or all users. To set environment variables for all users, you need to have administrator privileges.
5. In the **Variable name** field, enter the name of the environment variable you want to set. For a list of the environment variables that Nuke understands, see "Nuke Environment Variables" on page 614.
6. In the **Variable value** field, enter the value for the variable. The value can be a directory path, for example.
7. Click **OK**.

Note *When editing existing system variables or adding or deleting either user or system variables on Windows Vista, you need to log off and on again before your changes to environment variables take effect.*

On Mac

On Mac OS X, there is a special environment file - a .plist or property list file - which is read every time a user logs in. You may need to create the .plist file if it doesn't already exist in ~/.MacOSX/environment.plist (where "~" indicates the user's home directory, and "." a hidden directory).

Environment variables set using the .plist file are read both when Nuke is launched from the Nuke icon and when it's launched from the Terminal.

1. Open a Terminal window. By default, you should be in your home directory (your own directory in the Users folder). Enter **pwd** (present working directory) to verify this.
2. Enter **`ls -al`** to see a list of files in that directory. If .MacOSX is not in the list, enter **`mkdir .MacOSX`** to create the directory.
3. To create your .plist file, launch TextEdit.
4. Copy the following into the document and edit the "key" and "string" entries:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//
EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>NUKE_PATH</key>
    <string>/SharedDisk/Nuke</string>
    <key>OFX_PLUGIN_PATH</key>
    <string>/SharedDisk/OFX</string>
</dict>
</plist>
```

   This example sets two environment variables: NUKE_PATH and OFX_PLUGIN_PATH. NUKE_PATH points to /SharedDisk/Nuke/, and OFX_PLUGIN_PATH to /SharedDisk/OFX.

   For a list of the environment variables that Nuke understands, see "Nuke Environment Variables" on page 614.

5. Once you are happy with the document, select **Format** > **Make Plain Text**.
6. Save the file to your home directory with the name **environment.plist**. Make sure a .txt extension is not added to the end of the file name.
7. Quit TextEdit, and launch a Terminal window. Enter **pwd** to make sure you are in your home directory.
8. To move the environment.plist file from your home directory into the .MacOSX directory, enter **`mv environment.plist .MacOSX`**.
9. Log out and log in again.

On Linux

1. The procedure for setting an environment variable depends on what your default shell is. To get the name of the shell you are using, launch a shell and enter **`echo $SHELL`**.
2. Depending on the output of the previous step, do one of the following:
   - If your shell is a csh or tcsh shell, add the following command to the .cshrc or .tcshrc file in your home directory: **`setenv VARIABLE value`**. Replace **`VARIABLE`** with the name of the environment variable and **`value`** with the value you want to give it, for example **`setenv NUKE_PATH /SharedDisk/Nuke`**.
   - If your shell is a bash or ksh shell, add the following command to the .bashrc or .kshrc file in your home directory: **`export VARIABLE=value`**. Replace **`VARIABLE`** with the name of the environment variable and **`value`** with the value you want to give it, for example **`export NUKE_PATH=/SharedDisk/Nuke`**.

   For a list of the environment variables that Nuke understands, see "Nuke Environment Variables" on page 614.

To check if an environment variable exists:

On Windows

1. Select **Start** > **All Programs** > **Accessories** > **Command Prompt**.
2. In the command window that opens, enter **`echo %VARIABLE%`**. Replace **`VARIABLE`** with the name of the environment variable. For example, to check if NUKE_DISK_CACHE is set, enter **`echo %NUKE_DISK_CACHE%`**.

If the variable is set, its value is displayed in the command window.

On Mac or Linux

1. Launch Terminal or a shell.
2. Enter **`echo $VARIABLE`**. Replace **`VARIABLE`** with the name of the environment variable. For example, to check if NUKE_DISK_CACHE is set, enter **`echo $NUKE_DISK_CACHE`**.

If the variable is set, its value is displayed in the Terminal or shell window.

To display a list of set environment variables:

On Windows

1. Select **Start** > **All Programs** > **Accessories** > **Command Prompt**.
2. In the command window that opens, enter `set`.

A list of all the environment variables that are set is displayed in the command window.

On Mac or Linux

1. Launch Terminal or a shell.
2. Enter `printenv`.

A list of all the environment variables that are set is displayed in the Terminal or shell window.

Nuke Environment Variables

The following table lists the environment variables Nuke recognizes.

Environment Variable	Description
FOUNDRY_LICENSE_FILE	The location of the Nuke license file (a plain text file called nuke.lic), if the following recommended location is not used: On Mac OS X and Linux: /usr/local/foundry/FLEXlm On Windows XP: *drive letter*:\Program Files\The Foundry\FLEXlm OR *drive letter*:\Program Files (x86)\The Foundry\FLEXlm (for 32-bit Nuke on 64-bit Windows) On Windows Vista: *drive letter*:\ProgramData\The Foundry\FLEXlm
NUKE_PATH	The location where files related to Nuke customizations are stored. For more information, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
OFX_PLUGIN_PATH	The location where Nuke looks for OFX plug-ins. For more information, see "Loading OFX Plug-ins" on page 617.

Environment Variable	Description
NUKE_TEMP_DIR	The location where Nuke saves any temporary files that do not have a particular place defined for them. This is also where the **flipbook cache** setting in the Preferences defaults to.
NUKE_EXR_TEMP_DIR	On Linux, this is the location Nuke will use for temporary files while reading PIZ-compressed EXR files. This environment variable is only relevant on Linux. If this variable is not set, the location is determined by NUKE_TEMP_DIR.
NUKE_DISK_CACHE	The location where Nuke saves all recent images displayed in the Viewer. Ideally, this should be a local disk with the fastest access time available. It should also have enough space for the maximum flipbook cache size. If this variable is not set, the location is determined by the **flipbook cache** setting in the Preferences. By default, this setting points to NUKE_TEMP_DIR.
NUKE_DISK_CACHE_GB	The maximum size the flipbook cache can reach (in gigabytes). If this variable is not set, the location is determined by the **flipbook cache size** setting in the Preferences.
NUKE_DEBUG_MEMORY	When working on large images, Nuke may need to free up memory during rendering. When this happens and NUKE_DEBUG_MEMORY is set to 1, Nuke prints the following information to the console: *Memory: over maximum usage, trying to reduce usage from 1GB to 924MB.* If this variable is not set, you cannot see the debug memory messages. Note that here, KB, MB, GB, and TB mean units of 1000. For example, 1MB means 1,000,000 bytes.
FC_PATH	If you want to use a different version of FrameCycler than the one shipped with Nuke, you can use this to set the base directory of FrameCycler. This points to the FrameCycler binary.

Environment Variable	Description
FC_HOME	This points to the directory that contains bin/framecycler.
FC_DISK_CACHE	The location where Nuke saves all recent images flipbooked with FrameCycler. If this variable is not set, the location is determined by NUKE_DISK_CACHE.

Loading Gizmos, NDK Plug-ins, and TCL scripts

On start-up, Nuke scans various directories for files that customize the behavior of Nuke. It looks for information on favorite directories, menu options, image formats, gizmos, NDK plug-ins, generic TCL scripts, and preferences.

It looks in specific subdirectories of the user's home directory and the Nuke application directory as follows:

- **Linux:**
  /users/*login name*/.nuke
  /usr/local/Nuke6.2v2/plugins
- **Mac OS X:**
  /Users/*login name*/.nuke
  /Applications/Nuke6.2v2/Nuke6.2v2.app/Contents/MacOS/plugins
- **Windows:**
  In the .nuke directory, which can be found under the directory pointed to by the HOME environment variable. If this variable is not set (which is common), the .nuke directory will be under the folder specified by the USERPROFILE environment variable - which is generally of the form *drive letter*:\Documents and Settings\login name\ (Windows XP) or *drive letter*:\Users\login name\ (Windows Vista).

  To find out if the HOME and USERPROFILE environment variables are set and where they are pointing at, enter %HOME% or %USERPROFILE% into the address bar in Windows Explorer. If the environment variable is set, the folder it's pointing at is opened. If it's not set, you will get an error.

  drive letter:\Program Files\Nuke6.2v2\plugins (or, when using 32-bit Nuke on 64-bit Windows, *drive letter*:\Program Files (x86)\Nuke6.2v2\plugins)

If you want Nuke to look for plug-ins somewhere else rather than in these default locations, you can also define a common plug-in path yourself. Thus, by defining the Nuke plug-in path, you can assign yourself a common shared directory from which to control Nuke for multiple artists.

To define the Nuke plug-in path:

1. On each artist's machine, create an environment variable called **NUKE_PATH**.
2. Assign the **NUKE_PATH** environment variable to the pathname of the directory where you intend store files related to Nuke customizations.

For example, on Mac OS X:

```
setenv NUKE_PATH /SharedDisk/Nuke
```

Loading Python Scripts

On start-up, Nuke scans various directories for Python scripts that customize the behavior of Nuke. For example, the buttons on the toolbars are configured with the toolbars.py script.

It looks in specific subdirectories of the Nuke application directory as follows:

- **Linux:**
  /usr/local/Nuke6.2v2/plugins/nukescripts
- **Windows:**
  drive letter:\Program Files\Nuke6.2v2\plugins\nukescripts or
  drive letter:\Program Files (x86)\Nuke6.2v2\plugins\nukescripts
- **Mac OS X:**
  /Applications/Nuke6.2v2/plugins/nukescripts

Warning *It's worth saying that you should edit these files with care as mistakes could stop Nuke from running.*

Loading OFX Plug-ins

On start-up, Nuke scans various directories for OFX plug-ins that will bring additional functionality to Nuke. Paths to these directories vary between operating systems, but here are examples of where you may find them:

- **Linux:**
  /usr/OFX/Nuke
  /usr/OFX/Plugins
- **Windows:**

C:\Program Files\Common Files\OFX\Nuke (or, when using 64-bit Nuke on 64-bit Windows, \Program Files (x86)\Common Files\OFX\Nuke)

C:\Program Files\Common Files\OFX\Plugins (or, when using 64-bit Nuke on 64-bit Windows, \Program Files (x86)\Common Files\OFX\Plugins)

- Mac OS X:

  /Library/OFX/Nuke

  /Library/OFX/Plugins

If you want Nuke to look for OFX plug-ins somewhere else you can. Just define the environment variable OFX_PLUGIN_PATH to point to the new directory.

For example, on Mac OS X:

```
setenv OFX_PLUGIN_PATH /SharedDisk/OFX
```

Defining Common Favourite Directories

Favorite directories can be accessed by artists with a single click from any Nuke file browser. Typically you would create these favorites for common directories on a project.

To define a common set of favorite directories:

1. Create a file called **`menu.py`** in your plug-in path directory.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Add an entry in the following format:

   `nuke.addFavoriteDir('DisplayName', 'Pathname')`

   - Replace **`DisplayName`** with the string you want as the display name for the directory link, for example **`'Home'`** or **`'Desktop'`**.
   - Replace **`Pathname`** with the pathname to the directory to which the favorite should point.
3. In the above entry, you can also add the following optional arguments after **`'Pathname'`**:

   - **`type`**. This is an integer and a bitwise operator, OR a combination of **`nuke.IMAGE`**, **`nuke.SCRIPT`** or **`nuke.FONT`**.

     `nuke.IMAGE` restricts the favourite directory to appear only in the image file browser, which Nuke opens for file Read/Write operations.

     `nuke.SCRIPT` restricts the favourite directory to appear in the script file browser, which appears when you choose **File** > **Open** or a similar menu selection to open or import script files.

`nuke.FONT` restricts the favourite directory to appear in the font browser.

- **`icon='Name'.`** Replace **`Name`** with the name and file extension of the png (or xpm) image you wish to use as the icon for the favorite directory. This image must be stored in your Nuke plug-in path directory. It should be 24 x 24 pixels in size.
- **`tooltip='My tooltip'`**. Replace **`My tooltip`** with the string you wish to display as pop-up help.

Example 1

The following entry would create a favorite called **`DisasterFlickStore`** which appears on all File Browsers invoked from Read nodes and which points to the **`/job/DisasterFlick/img`** directory.

```
nuke.addFavoriteDir ('DisasterFlickStore', '/job/DisasterFlick/img', nuke.IMAGE)
```

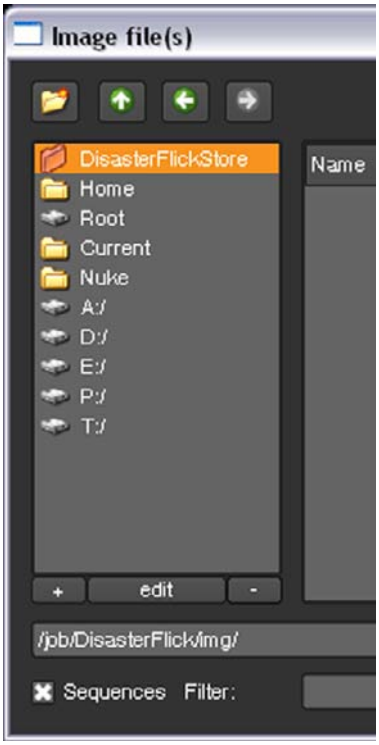


Image file(s)
DisasterFlickStore
Home
Root
Current
Nuke
A:/
D:/
E:/
P:/
T:/
Name
+
edit
-
/job/DisasterFlick/img/
Sequences
Filter:

Figure 21.2: The result of example 1.

Example 2

The following entry would create a favorite called **`Test`**. It appears on all File Browsers invoked from Read nodes or by selecting **File** > **Open** and points to the **`/job/Test`** directory. The entry also defines *Test Images and Scripts* as the tool tip for the favorite directory.

```
nuke.addFavoriteDir ('Test', '/job/Test', nuke.IMAGE |
nuke.SCRIPT, tooltip='Test images and Scripts')
```

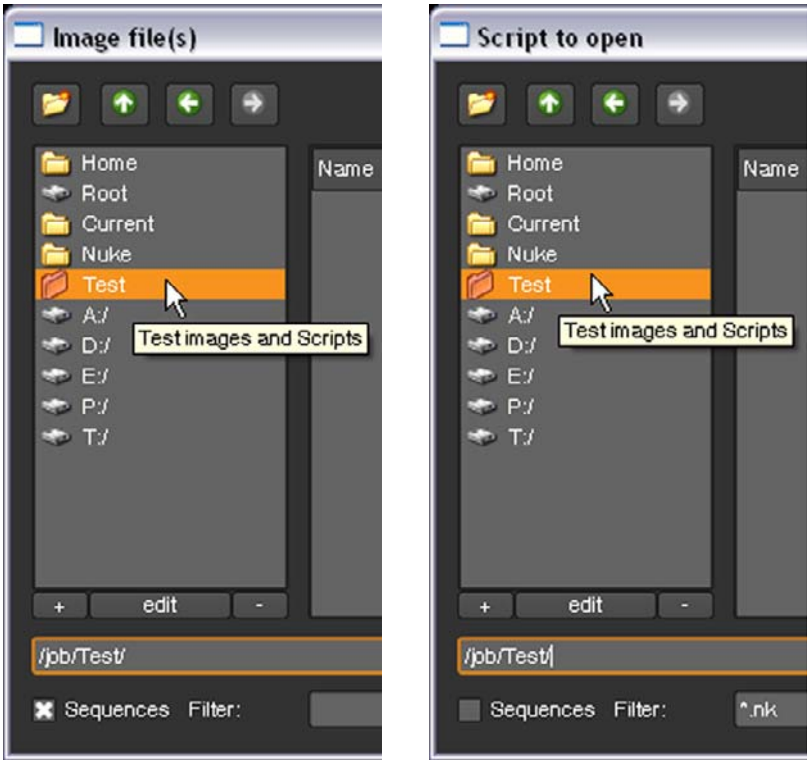


Image file(s)
Home
Root
Current
Nuke
Test
A:/
D:/
E:/
P:/
T:/
Test images and Scripts
Name
+
edit
-
/job/Test/
Sequences
Filter:
Script to open
Home
Root
Current
Nuke
Test
A:/
D:/
E:/
P:/
T:/
Test images and Scripts
Name
+
edit
-
/job/Test/
Sequences
Filter:
*.nk

Figure 21.3: The result of example 2.

Handling File Paths Cross Platform

If your facility uses Nuke on several operating systems, you may want to configure Nuke to replace the beginnings of file paths so that scripts created on one platform will also work on another.

For example, to ensure file paths created on Windows also work on Linux and vice versa, you can do the following:

1. Create a file called **`init.py`** in your plug-in path directory if one doesn't already exist.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. Open the init.py file in a text editor and add an entry in the following format:

```
import platform


def filenameFix(filename):
   if platform.system() in ("Windows", "Microsoft"):
      return filename.replace( "/SharedDisk/", "p:\\" )
   else:
      return filename.replace( "p:\\", "/SharedDisk/" )
   return filename
```

   This way, the Windows file paths (beginning with **p:\** in the above example) will be replaced with the Linux file paths (beginning with **/SharedDisk/**) under the hood whenever a Nuke script is used on Linux. Otherwise, the Windows file paths are used.

   Note that the file paths displayed in the graphical user interface (GUI) will not change. If you are using **p:\** in a node control, this will still be displayed as **p:\**. However, on Linux, Nuke will interpret **p:\** as **/SharedDisk/**.

Defining Custom Menus and Toolbars

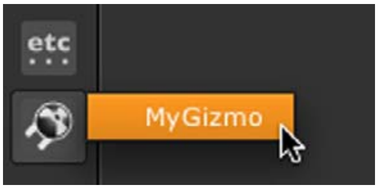
You can freely add custom menus and menu options as well as toolbars and toolbar options to the Nuke interface. Artists can then use these options to trigger gizmos and plug-ins stored in the plug-in path directory.

For example, to add a new menu in the default Toolbar with an option to trigger a gizmo called MyGizmo, you can do the following:

1. In your home directory, create a directory called .nuke (if it doesn't exist already). For more information on this directory, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. In the .nuke directory, create a file called `menu.py` if one does not already exist.
3. In a text editor, modify the file `menu.py`, adding the lines:

```
toolbar = nuke.toolbar("Nodes")
toolbar.addCommand( "Test/MyGizmo", "nuke.createNode('MyGizmo')")
```

   This adds a menu labeled "Test" to the default Nodes Toolbar with an item labeled "MyGizmo" that creates an instance of the node MyGizmo.



It's also possible to add items to other menus in Nuke and even create your own toolbars. The following sections cover these possibilities in detail.

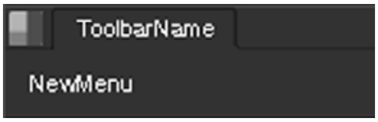
To add a toolbar:

1. Create a file called **`menu.py`** in your plug-in path directory if one doesn't already exist.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Open the menu.py file in a text editor and add an entry in the following format:

   ```
   t=nuke.toolbar("ToolbarName")
   t.addCommand("NewMenu", "PythonCode", "Shortcut",
   icon="IconName")
   ```

   - Replace **`ToolbarName`** with the name you want to give to the toolbar. This name will appear in the content menus under "Pane" and above the toolbar on the title tab.
   - Replace **NewMenu** with the name of the menu you want to add to the toolbar. The following image illustrates where whatever you use to replace **ToolbarName** and **NewMenu** will appear in the new toolbar.



   - Replace **`PythonCode`** with relevant Python code (usually **`nuke.createNode`**), and, if necessary, use the name of the gizmo, generic Python script, or plug-in file you want the menu option to invoke. For ease of use, place all such referenced files inside the plug-in path directory.

     If you like, you can also replace **`PythonCode`** by a Python callable.
   - Replace **`Shortcut`** with a keyboard shortcut, for example **Alt+A**, **Ctrl/Cmd+A**, or **Shift+A**. The letter **a** alone represents lower-case **a**. **F1** represents function key 1. You can combine the **Shift**, **Ctrl/Cmd**, and **Alt** keys as necessary. If you like, you can also use **#A** to represent **Alt+A,** **^A** to represent **Ctrl/Cmd+A**, and **+A** to represent **Shift+A**.

   - Replace `IconName` with the name of the png (or xpm) image you wish to use as the menu icon. This image must be stored in your Nuke plug-in path directory. It should be 24 x 24 pixels in size.

3. In the above entry, you can also add the following optional arguments in the parenthesis after `"ToolbarName"`:
   - `True`. This is the default. When True, nuke.toolbar() calls the toolbar with the given name or creates it if it does not exist. For example, `t=nuke.toolbar("Extras", True)` would either call an existing toolbar called Extras or create one if it did not already exist.
   - `False`. When False, the toolbar is not created if it does not already exist and nuke.toolbar() returns **None**. You can use this to find out if a toolbar with a given name already exists. For example, `t=nuke.toolbar("Extras", False)` would either call an existing toolbar called Extras or return None if such a toolbar did not exist.

The new toolbar does not appear by default, but is listed under **Pane** in the content menus. From there, you can insert it in any pane. Once you are happy with the new toolbar and its position, save the layout (select **Layout** > **Save Layout 1**). Thereafter, the toolbar will appear whenever Nuke is launched.

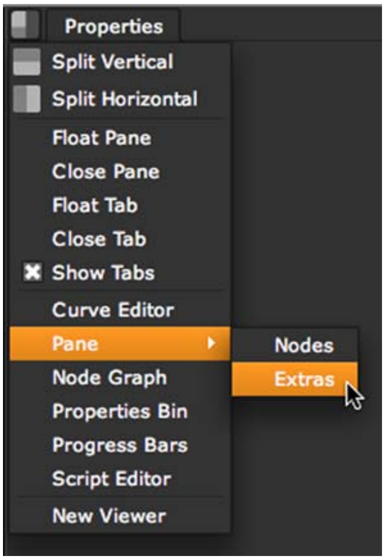

Properties
Split Vertical
Split Horizontal
Float Pane
Close Pane
Float Tab
Close Tab
Show Tabs
Curve Editor
Pane
Node Graph
Properties Bin
Progress Bars
Script Editor
New Viewer
Nodes
Extras

Figure 21.4: Custom toolbars are listed under "Pane" in the content menus.

You can build several toolbars for different tasks and save layouts that have one or another present for easy context switching.

Example 1

The following entry creates a new toolbar called **Extras**. The toolbar includes an option called **Create VectorBlur** that creates a VectorBlur node. The entry also defines **v** as the keyboard shortcut for the VectorBlur node.

```
t=nuke.toolbar("Extras")
t.addCommand("Create VectorBlur", "nuke.createNode ('VectorBlur')", "v")
```

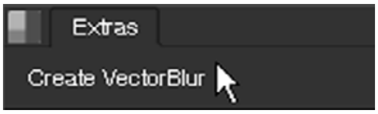


Figure 21.5: The result of example 1.

Example 2

In this example, we add an option called Autoplace to the toolbar created in example 1. This option places the selected nodes neatly one after another, as illustrated in the following images:

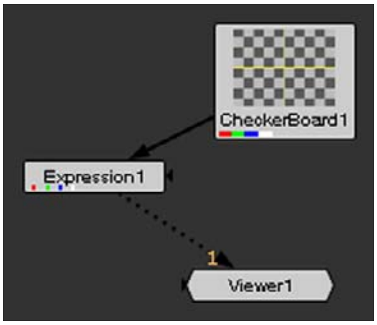


Figure 21.6: Using Autoplace to tidy up the Node Graph: Before Autoplace.

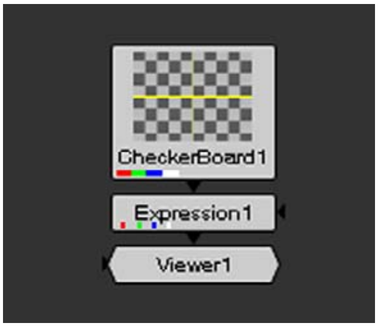


Figure 21.7: Using Autoplace to tidy up the Node Graph: After Autoplace.

The following entry adds the Autoplace option. It also defines **Alt+A** as the keyboard shortcut for this option.

```
def _autoplace():
  n = nuke.selectedNodes()
  for i in n:
    nuke.autoplace(i)
t=nuke.toolbar("Extras")
t.addCommand("Auto&place", "_autoplace()", "Alt+a")
```

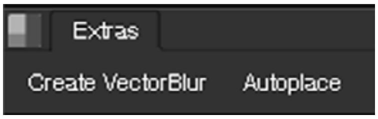


Figure 21.8: The result of example 2.

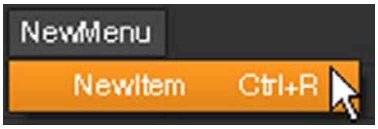
To define a menu or toolbar option:

1. If you haven't already done so, create a file called **`menu.py`** in your plug-in path directory. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Open the menu.py file in a text editor and add an entry in the following format:

```
menubar=nuke.menu("MenuType")
m=menubar.addMenu("&NewMenu")
m.addCommand("&NewItem", "PythonCode", "Shortcut",
icon="IconName", index=#)
```

   - Replace **`MenuType`** with the type of menu or toolbar you want to add an item to:

     `Nuke` adds an item to the application main menu bar.

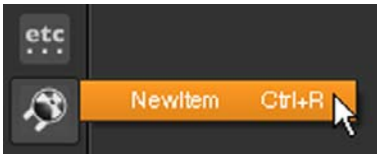


`Animation` adds an item to the pop-up menu on the Animation button of all panels, and to the right-click pop-up menu of the Curve editor.

`Properties` adds an item to the right-click menus of properties panels.

`Node Graph` adds an item to the right-click menu of the Node Graph.

`Nodes` adds an item to the default Toolbar.



`Viewer` adds an item to the right-click menu of the Viewer.

`Pane` adds an item to the content menus where it will appear under **Pane**.

- Replace **`NewMenu`** with the menu name. Using an existing menu name appends any new options to the existing menu. You can also add options to the default Menu bar and Toolbar.
- Replace **`NewItem`** with the underlying item you want to add to the menu. You may precede any character with an **&** in order to flag it as hotkey trigger.
- Replace **`PythonCode`** with relevant Python code (usually **`nuke.createNode`**) and, if necessary, use the name of the gizmo, generic Python script, or plug-in file you want the menu option to invoke. For ease of use, place all such referenced files inside the plug-in path directory.

  For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

  If you like, you can also replace **`PythonCode`** by a Python callable. This has the advantage that you get informed about errors in your script on start-up instead of when the menu item is invoked. For an example of using a lambda function, see "Example 3" on page 628.
- Replace **`Shortcut`** with a keyboard shortcut, for example **Alt+A**, **Ctrl/Cmd+A**, or **Shift+A**. The letter **a** alone represents lower-case **a**. **F1** represents function key 1. You can combine the **Shift**, **Ctrl/Cmd**, and **Alt** keys as necessary. If you like, you can also use **#A** to represent **Alt+A, ^A** to represent **Ctrl/Cmd+A**, and **+A** to represent **Shift+A**.

Note *By assigning a keyboard shortcut, you can overwrite existing shortcuts. For example, if you assign the shortcut **Ctrl/Cmd+O** to a new menu item, it will no longer be used for it's default purpose which is opening a file. However, shortcuts are only overwritten in the main menu bar, the Toolbar, any user-created toolbars, and the menu you are adding the new menu item to. This means you can add a shortcut into the Node Graph, for example, without resetting the same shortcut in the Viewer. However, you cannot add a shortcut into the Node Graph without resetting the same shortcut in the main menu bar or the Toolbar.*

- Replace **`IconName`** with the name of the png (or xpm) image you wish to use as the menu icon. This image must be stored in your Nuke plug-in path directory. It should be 24 x 24 pixels in size.
- Replace # with a number that represents the position of the item in the menu or toolbar. If you do not use an index keyword, the item will be added in the end of the menu or toolbar.

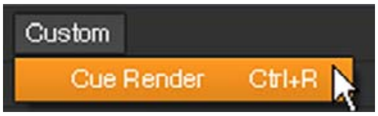
Tip *You can also put the menu name in the addCommand call, like this:*

```
nuke.menu("MenuType").addCommand("NewMenu/NewItem",
"PythonCode("name")")
```

Example 1

The following entry creates a new menu and option called **Custom** > **Cue Render** in the menu bar. It inserts a gizmo called "cue_render." The entry also defines **Ctrl+R** as the keyboard shortcut for the gizmo.

```
menubar=nuke.menu("Nuke")

m=menubar.addMenu("&Custom")

m.addCommand("&Cue Render", "nuke.createN-
ode('cue_render')", "Ctrl+R")
```



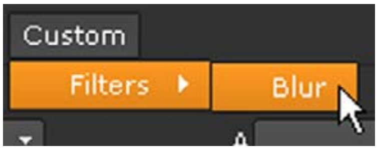
Example 2

For information on how to create a new menu in the default Toolbar with a menu item that triggers a gizmo, see the example on page 622.

Example 3

The following entry creates a menu and options called **Custom** > **Filters** > **Blur** in the menu bar. Selecting **Blur** inserts the Blur node.

```
menubar=nuke.menu("Nuke")

m=menubar.addMenu("&Custom")

m.addCommand("Filters/Blur", "nuke.create-
Node(\"Blur\")" )
```



You can also do the same with a lambda function:

```
menubar=nuke.menu("Nuke")

m=menubar.addMenu("&Custom")

m.addCommand("Filters/Blur", lambda: nuke.create-
Node("Blur") )
```

This way, you don't have to use the backslashes.

Defining Common Image Formats

You may wish to define image formats (image resolutions and corresponding pixel aspect ratios) for a given project. These will appear as pulldown options on Read and Reformat nodes.

To define an image format:

1. If you haven't already done so, create a file called **menu.py** in your plug-in path directory.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Add an entry in the following format:

   ```
   nuke.addFormat(" ImageWidth ImageHeight LowerLeftCorner LowerRightCorner UpperRightCorner UpperLeftCorner PixelAspectRatio DisplayName ")
   ```

   - Replace `ImageWidth` with the width (in pixels) of the image format.
   - Replace `ImageHeight` with the height (in pixels) of the image format.
   - If you wish to define an image area that is smaller than the format resolution, replace `LowerLeftCorner, LowerRightCorner, UppperRightCorner, UpperLeftCorner` with the proper x and y coordinates (in pixels) for the lower left, lower right, upper right, and upper left corners of this smaller image area (optional).
   - Replace `DisplayName` with display name of the format. This will appear on all relevant pulldown menus.

Example 1

The following entry would create a new format called `full_aperature_anamorphic` with a resolution of 2048 by 1556 and a pixel aspect ratio of 2.0. (As the image corners are not explicitly defined, the image will cover the entirety of the format area.)

```
nuke.addFormat (" 2048 1556 2.0 full_aperture_anamorphic ")
```

Gizmos, Custom Plug-ins, and Generic TCL Scripts

Nuke allows artists and technical directors to create *gizmos*, which are simply groups of Nuke nodes that may be reused by other artists. These are equivalent to Shake's macros. Studios commonly use gizmos to consistently apply certain color grading techniques, process incoming footage according to a particular conversion formula, and process outgoing footage in preparation for film printing.

A gizmo is a Group Node that you create and save in a separate .gizmo file in your Nuke plug-in folder. Nuke scripts can use this gizmo just like any other node type. Saved scripts only contain the name and control settings for the gizmo; the definition is in the gizmo file and it is read at the same time the script is loaded into Nuke. Thus, you can alter the implementation of the gizmo and change all the scripts that are using it.

Note *Unlike other nodes, gizmos cannot be cloned. For more information on cloning nodes, see "Cloning Nodes" on page 51.*

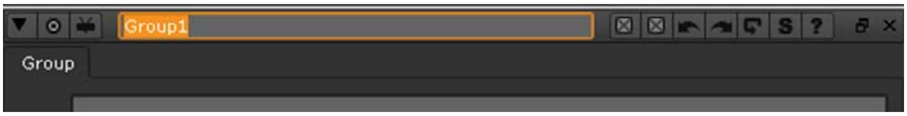
Creating and Sourcing Gizmos

Using Nuke's **Export gizmo** command, you can export a group of nodes and explicitly control which controls may be edited by the artists to ensure the processes within the gizmo are consistently applied.

Creating gizmos

To create a gizmo:

1. Select the nodes you want to include in the gizmo.
2. Select **Other** > **Group** from the Toolbar (or press **Ctrl/Cmd+G**) to group the nodes.
3. You may want to rename the group by entering a new name in the Group properties panel title field. This step is optional, and has no effect on the saved gizmo. However, it is often a good idea to give the Group the same name you intend to use for the gizmo.



4. To control which controls the artists can adjust, follow the instructions under *Managing the gizmo controls* below.
5. Click the **export as gizmo** button.
6. In the file browser that appears, click **Home**. Type **`.nuke/`** after the path displayed at the bottom of the file browser.
7. Enter a name after the path, and append a **`.gizmo`** extension after the name. This is the name of the command that will be written to any saved script that's using the gizmo. The name should begin with a capital letter, because Nuke uses this as an indication that the command is a node or a gizmo. Thus, it will produce a more useful error message if the gizmo file is not found when the script is loaded.
8. Click **Save**.

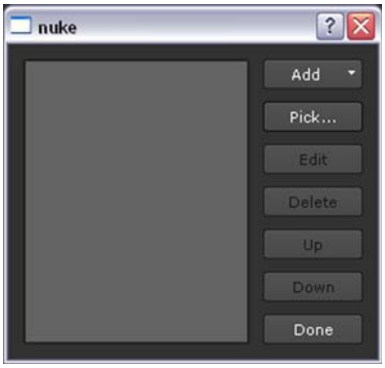
Managing the gizmo controls

You can add controls to your gizmo (which at this point is just a Group node) in two different ways:

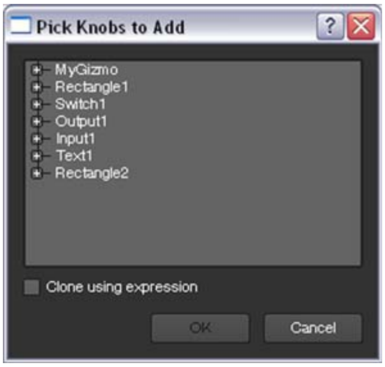
- by picking and editing a control from the controls that by default exist for the nodes inside your Group node. For example, if the Group node contains a Grade node, you can add any of the Grade node controls to your gizmo properties panel.
- by adding a control you have created yourself to your gizmo properties panel.

To pick existing controls:

1. Right-click on the dark gray background of the Group properties panel and select **Manage User Knobs**. The following dialog opens.

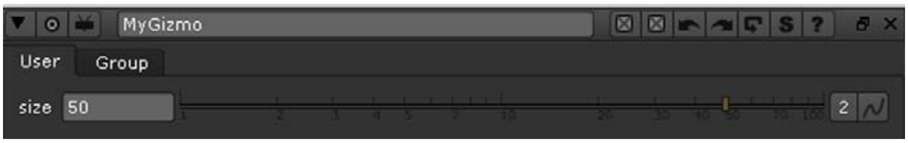

nuke
Add
Pick...
Edit
Delete
Up
Down
Done

2. To pick a control you want to give the users control of, click on the **Pick** button. This opens a dialog that lists all the nodes that the group contains.


Pick Knobs to Add
MyGizmo
Rectangle1
Switch1
Output1
Input1
Text1
Rectangle2
Clone using expression
OK
Cancel

Expand the list items as necessary to see the controls you can include in the gizmo controls. Select a control and click **OK**. You can also select multiple controls by **Ctrl/Cmd**+clicking on them, or pick a range of controls by **Shift**+clicking.

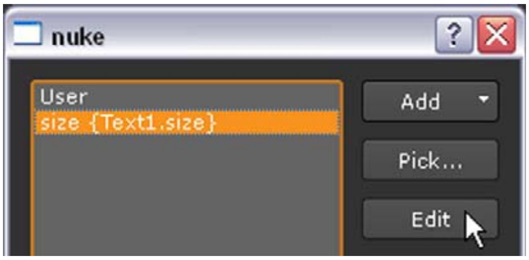
At this point, a new tab called **User** appears in the Group properties panel. The control you selected has been added to this tab. In our example, we selected the **size** control of the Text node.



size 50

The control has also been added to the Manage User Knobs dialog, ready for you to edit.

3. To edit the control you added, open the Manage User Knobs dialog and select the control from the list. Click **Edit**.



User
size {Text1.size}

In most cases, you can edit the following:

- **Name** - Give the new control a unique name here. You need to use this name whenever you want to reference the control from scripts or via expressions. The name can only contain letters and digits. Spaces or punctuation are not allowed. This field cannot be left empty.
- **Label** - Whatever you enter here appears to the left of the control in the gizmo properties panel (or, in the case of buttons, on the button). If you leave this empty, whatever is in the **Name** field is also used as the label.

  In the **Label** fields of check boxes, TCL script buttons, and Python script buttons, you can also use HTML. For example, to have your text appear in bold, you can enter **text**.

  To add an icon to your check box or TCL/Python button using HTML, you can enter **** in the **Label** field. This adds the Nuke color wheel icon. You can also use your own icons in the same way as long as you save them in your plug-in path directory. Most common image formats will work, but we recommend using PNG files.

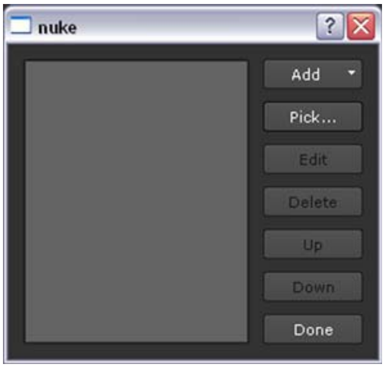
Note that the HTML has been changed to a slightly non-standard form where newlines are significant. If there is a newline character in your data, a new line will be displayed in the label.

- **Tooltip** - Enter a short help text here. It will appear, along with the Name, in a pop-up tool tip when the user points the mouse at the control. If you do not provide a tool tip, whatever is in the **Name** field will be used as the tool tip.
- **Hide** - Check this to hide the control from the users. This can be useful if you want to make a new control to contain a complex expression that you can then refer to repeatedly by other controls.
- **Start new line** - Uncheck this if you want the control to appear on the same line as the previous control in the gizmo properties panel.

4. If necessary, repeat the previous three steps to add more controls to your Group node (future gizmo).
5. In the Group node properties panel, the controls are listed in the same order as they are in the Manage User Knobs dialog. To move a control up or down in the properties panel, select it in the dialog and use the **Up** and **Down** buttons.
6. Once you have all the controls you need, proceed to step 5 under "Creating gizmos" on page 630.

To create new controls:

1. Right-click on the dark gray background of the Group properties panel and select **Manage User Knobs**. The following dialog opens.



2. To add a new control, tab, static text, or divider line to the Group (gizmo) controls, click **Add** on the Manage User Knobs dialog and select the option you want to add. This opens a dialog where you can edit the control, tab or static text you added. In most cases, you can edit the following:

- **Name** - Give the new control a unique name here. You need to use this name whenever you want to reference the control from scripts or via expressions. The name can only contain letters and digits. Spaces or punctuation are not allowed. This field cannot be left empty.
- **Label** - Whatever you enter here appears to the left of the control in the gizmo properties panel. If you leave this empty, whatever is in the **Name** field is also used as the label.

  In the **Label** fields of check boxes, TCL script buttons, and Python script buttons, you can also use HTML. For example, to have your text appear in bold, you can enter text.

  To add an icon to your check box or TCL/Python button using HTML, you can enter in the **Label** field. This adds the Nuke color wheel icon. You can also use your own icons in the same way as long as you save them in your plug-in path directory. Most common image formats will work, but we recommend using PNG files.

  Note that the HTML has been changed to a slightly non-standard form where newlines are significant. If there is a newline character in your data, a new line will be displayed in the label.
- **Tooltip** - Enter a short help text here. It will appear, along with the Name, in a pop-up tool tip when the user points the mouse at the control. If you do not provide a tool tip, whatever is in the **Name** field will be used as the tool tip.
- **Hide** - Check this to hide the control from the users. This can be useful if you want to make a new control to contain a complex expression that you can then refer to repeatedly by other controls.
- **Start new line** - Uncheck this if you want the control to appear on the same line as the previous control in the Group properties panel.

3. Use an expression to link the control you just created to a node and its control inside the Group node. This is important, because for the new control to do anything, you need to refer to it using an expression in some other control on a node inside the Group. For more information, see the examples below and refer to "Expressions" on page 533.
4. If necessary, repeat the previous four steps to add more controls to your Group node (future gizmo).
5. In the Group node properties panel, the controls are listed in the same order as they are in the Manage User Knobs dialog. To move a control up or down in the properties panel, select it in the dialog and use the **Up** and **Down** buttons.
6. Once you have all the controls you need, proceed to step 5 under "Creating gizmos" on page 630.

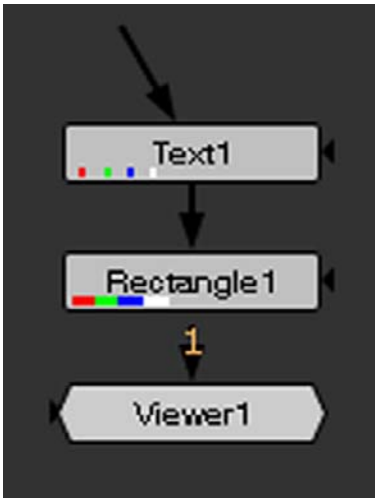
To delete controls:

1. Right-click on the dark gray background of the Group properties panel and select **Manage User Knobs**.
2. In the dialog that opens, select the controls that you want to delete from the list and click **Delete**.
3. To delete an entire tab, select all controls on the tab as well as the tab name and click **Delete**.

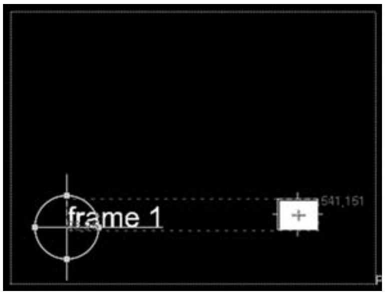
Examples

Below are some examples on how to create new controls for gizmos. To try them out, do the following preparations:

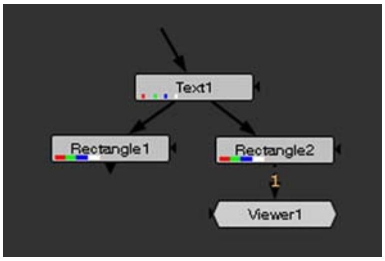
1. Select **Draw** > **Text** and **Draw** > **Rectangle**. Create the following setup:



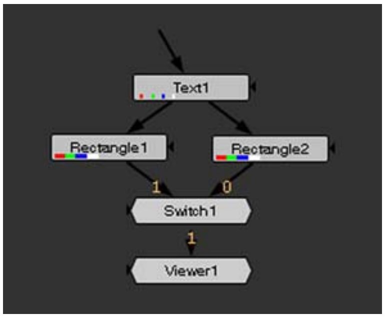
2. Double-click on the Rectangle1 node.
3. In the Viewer, resize and reposition the rectangle until it looks like the following:



4. In the Rectangle1 properties panel, go to the **Color** tab. Click on the **4** button to display multiple values rather than the slider. Enter **1** as the value for **r**, and 0 as the value for **b**, **g** and **a**. This changes the color of the rectangle from white to red.
5. Copy the Rectangle1 node and paste it into the same script. Create the following connections:



6. Double-click on the Rectangle2 node and change the color of the rectangle from red to green (r 0, g 1, b 0, a 0).
7. Select **Merge** > **Switch** to add a Switch node. Create the following connections:



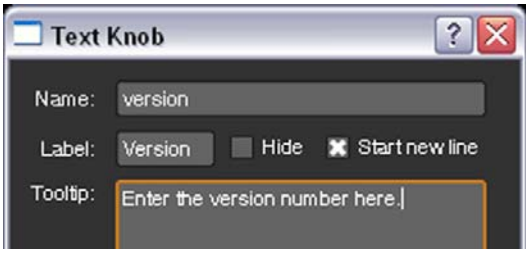
8. Select the Text1, Rectangle1, Rectangle2 and Switch1 nodes and press **Ctrl/Cmd+G** to group them. This group will be the gizmo we will add controls to in the following examples.
9. Delete the original four nodes from the **Node Graph** tab.
10. Select the Group node and append a Viewer to it.

Example 1

In this example, we add a control called **Version** to the Group node controls. This control is an input field. Whatever is entered in the field is called by the Text1 node and displayed in the Viewer when you view the output of the group.

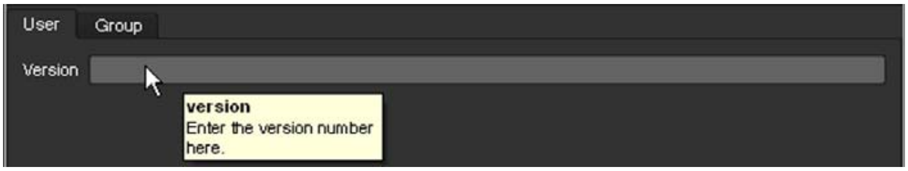
1. Open the Group properties panel and right-click on the dark gray background. Select **Manage User Knobs**.

2. In the dialog that opens, select **Add** > **Text input Knob** to add a text input field control to your Group properties panel.
3. Enter *version* as the **Name** for the control, *Version* as the **Label**, and *Enter the version number here* as the **Tooltip**. Click **OK** and **Done** to close the dialogs.



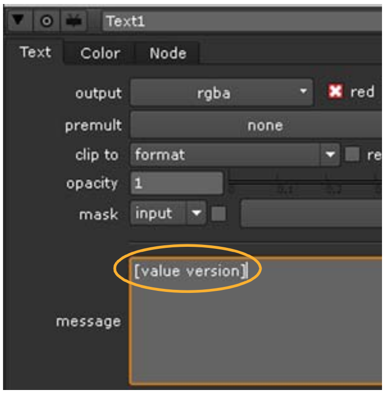
Name: version
Label: Version ☐ Hide ☒ Start new line
Tooltip: Enter the version number here.

This step created a tab called **User** in the Group node controls. All the controls you add or pick are added on this tab by default. As you can see, the **Version** control is now there.



version
Enter the version number here.

4. On the **Group1 Node Graph** tab, double-click the Text1 node to open its controls. In the **message** field, enter the following expression: **[value version]**. This expression calls the control named *version* that you created in the previous step. Therefore, whatever is entered in the **Version** field of the Group node (for example, **v03**), will appear as a result of the Text1 node.



output rgba ☒ red

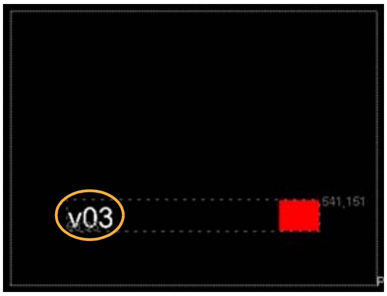
premult none

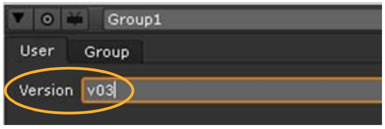
clip to format

opacity 1

mask input

message [value version]





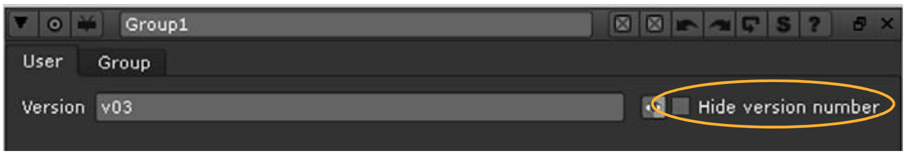
Version v03

Example 2

This example teaches you to create a checkbox control that the users can use to specify whether they want to display or hide the version number added in the previous example.

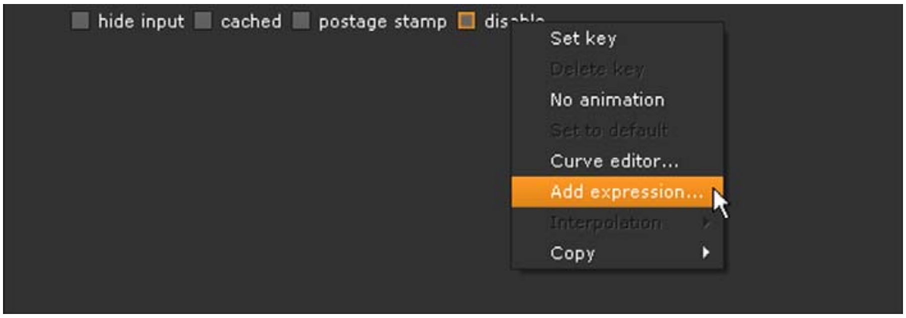
1. In the Group properties panel, right-click on the dark gray background and select **Manage User Knobs**.
2. In the dialog that opens, select **Add** > **Check Box** to add a checkbox control to your Group properties panel.
3. Enter *hideversion* as the **Name** for the control, *Hide version number* as the **Label**, and *Check this to hide the version number* as the **Tooltip**.
4. To have the new control appear next to the Version control (created in the previous example) rather than below it on its own line, uncheck **Start new line**. Click **OK** and **Done** to close the dialogs.

   The control you created appears in the Group properties panel now.



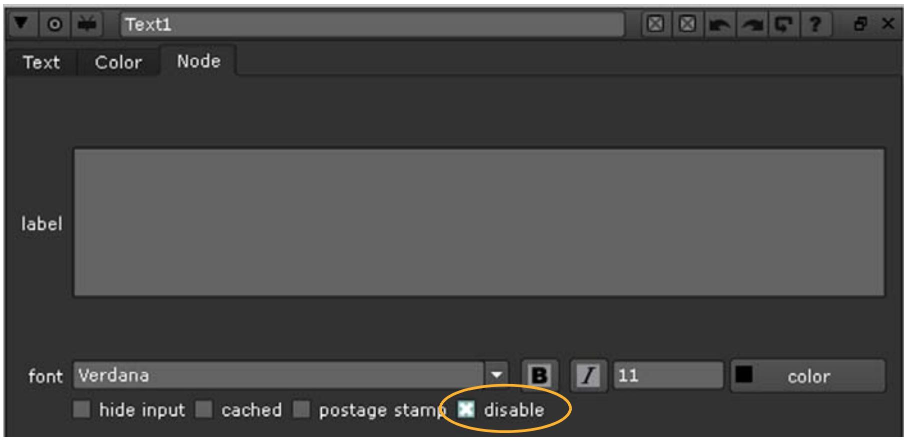
Version v03 ☐ Hide version number

5. In the Text1 controls, go to the **Node** tab. Right-click on the **disable** control and select **Add expression**.

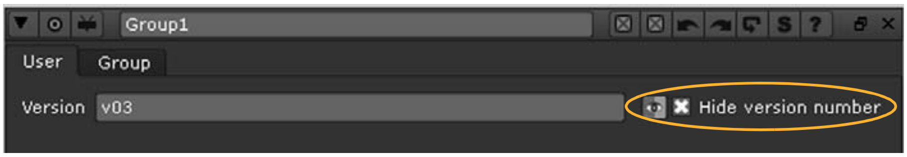

☐ hide input ☐ cached ☐ postage stamp ☐ disable
Set key
Delete key
No animation
Set to default
Curve editor...
Add expression...
Interpolation
Copy

6. In the **Expression** field, enter **hideversion** (or, if you want to make it clear that the control is in the enclosing group, you can also use **parent.hideversion**). This calls the control you created in steps 2 and 3. Click **OK**.

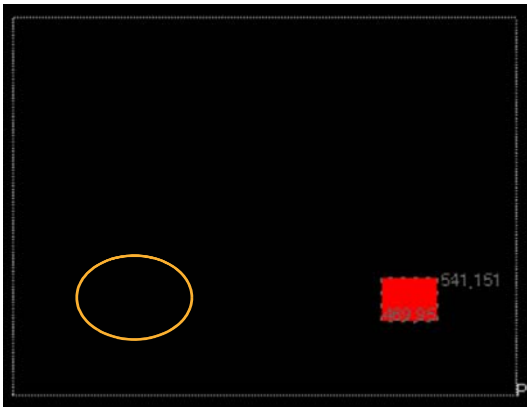
   From now on, whenever **Hide version number** is checked in the Group controls, the Text1 node is disabled and you cannot see the version number it would otherwise create.


Text1
Text
Color
Node
label
B
I
color

font Verdana, size 11
☐ hide input ☐ cached ☐ postage stamp ☒ disable


Group1
User
Group

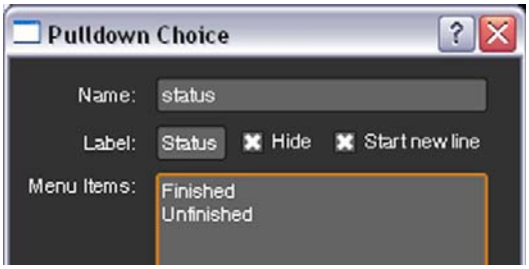
Version v03
☒ Hide version number



Example 3

In this example, we add we add a control labeled **Status** to the Group controls. This control is a pulldown menu with two options: **Finished** and **Unfinished**. When **Finished** is selected, the green rectangle is displayed. When **Unfinished** is chosen, you'll see the red rectangle instead.

1. In the Group properties panel, right-click on the dark gray background and select **Manage User Knobs**.
2. In the dialog that opens, select **Add** > **Pulldown Choice** to add a pulldown menu control to your Group properties panel.
3. Enter *status* as the **Name** for the control and *Status* as the **Label**. In the **Menu Items** field, list the items you want to appear in the pulldown menu - in this case, *Finished* and *Unfinished*.



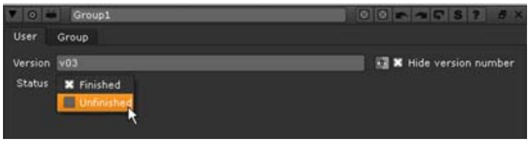
Pulldown Choice

Name: status

Label: Status ☒ Hide ☒ Start new line

Menu Items:
Finished
Unfinished

Finally, enter *Select the production status here* as the **Tooltip**. Click **OK** and **Done** to close the dialogs. The **Status** control should now appear in the Group controls.



Version v03 ☒ Hide version number

Status
☒ Finished
☐ Unfinished

4. On the **Group1 Node Graph** tab, double-click the Switch1 node to open its controls. Right-click on the **which** field and select **Add expression**.
5. In the dialog that opens, enter the following expression: status==0 (or, parent.status==0). This expression calls the control named *status* that you created earlier in this example. For the pulldown menus, the first item is 0, the next 1, the next 2, and so on.

   From now on, whenever **Finished** is selected under **Status**, the green rectangle is shown. When **Unfinished** is chosen, the red rectangle is shown.

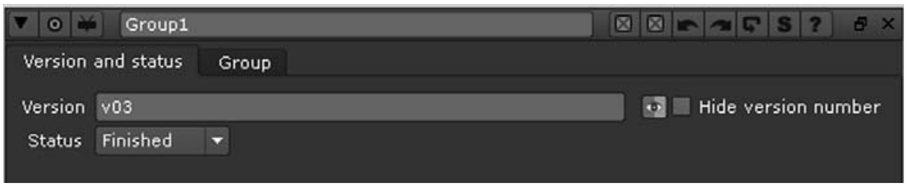
Example 4

This example teaches you how to visually group and rearrange the controls you created for the Group properties panel. You can do this by renaming the **User** tab, and using static text and divider lines to group the controls on the tab.

First, we'll rename the **User** tab in the Group properties panel:

1. In the Group properties panel, right-click on the dark gray background and select **Manage User Knobs**.
2. In the dialog that opens, select **User** and click **Edit**.
3. In the **Label** field, enter a new name for the tab, for example, *Version and status*. Click **OK** and **Done** to close the dialogs.

   If you now look at the Group controls, you'll notice that the **User** tab has been renamed to **Version and status**.



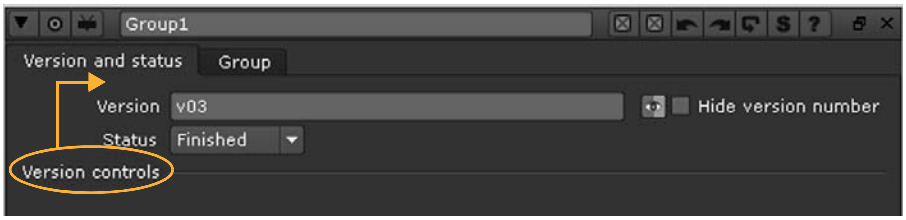
Version v03

Status Finished

Next, we'll group the two version controls of the Group node under a title called *Version controls*:

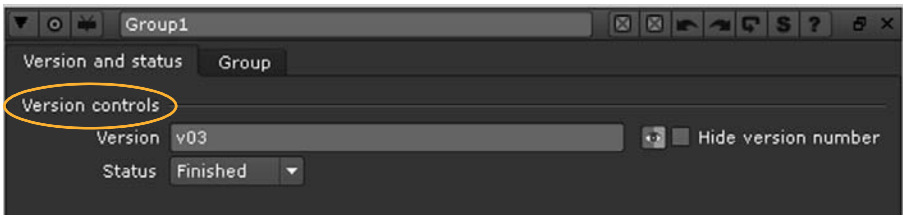
1. In the Group properties panel, right-click on the dark gray background and select **Manage User Knobs**.
2. In the dialog that opens, select **Add** > **Text** to add text to your Group properties panel.
3. Enter *versioncont* as the **Name** for the control and *Version controls* as the **Label**. Click **OK** and **Done** to close the dialogs.

This adds the text *Version controls* to the Group properties panel. However, the text does not appear where we want it to appear: on top of the **Version** and **Hide version number** controls. Let's move it up.


Group1
Version and status
Group
Version
v03
Hide version number
Status
Finished
Version controls

4. Right-click on the Group properties panel again and select **Manage User Knobs**.
5. Select **[Version controls]** from the list and click **Up** three times. Click **Done**.

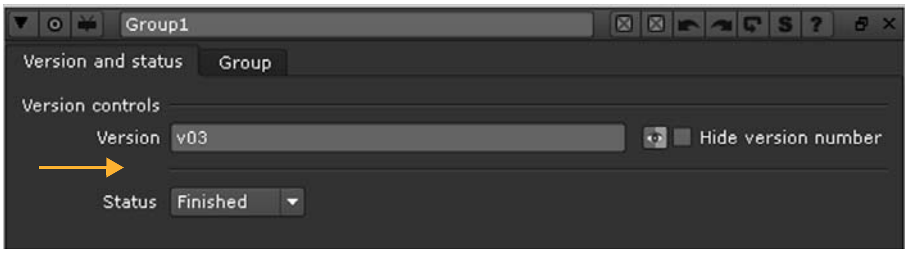
   The text should now appear on top of the Group properties panel, above the version controls.


Group1
Version and status
Group
Version controls
Version
v03
Hide version number
Status
Finished

Finally, we'll add a divider line between the version controls and the **Status** control:

1. In the Group properties panel, right-click on the dark gray background and select **Manage User Knobs** again.
2. In the dialog that opens, select **Add** > **Divider Line** to add a line to divide the controls in your Group properties panel.
3. Select the line from the Manage User Knobs dialog, where it is shown as **unnamed**.
4. Click the **Up** button once to move the line between the **Hide version number** and **Status** controls. Click **Done**.

   If you now open the Group controls, you'll notice that a there's a line between these controls.



Version v03

Status Finished

Hopefully, the above examples have given you an idea of how to create user controls for your gizmos. Once you have all the controls you need, remember to save your gizmo (for instructions, see step 5 under "Creating gizmos" on page 630).

Sourcing gizmos

To source a gizmo:

Create a menu option referencing the gizmo (see "Defining Custom Menus and Toolbars" on page 622),

OR

Instruct artists to invoke the gizmo by:

- typing **x** on the Node Graph or Properties Bin and entering the gizmo name (without the extension) as a TCL command in the dialog that opens.
- opening the Script Editor and entering **nuke.load ("*gizmo name*")** where ***gizmo name*** stands for the name of the gizmo without the extension.
- selecting **Other** > **All plugins** > **Update** and once this is done pressing **Tab** on the Node Graph and entering the gizmo name.

Custom Plug-ins

The Nuke developer's kit (NDK) allows developers to create and compile their own binary plug-ins.

To source a custom plug-in:

1. Place the plug-in file in the plug-in path directory. Its name should include a **`.dll`** (on Windows), **`.so`** (on Linux) or **`.dylib`** (on Mac) extension.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.

2. Create a menu option referencing the plug-in file (see "Defining Custom Menus and Toolbars" on page 622).

   Or instruct artists to invoke the plug-in by opening the Script Editor and entering nuke.load ("*plug-in name*") where *plug-in name* stands for the name of the plug-in without the extension.

Sourcing TCL Procedure

A Nuke script or gizmo is in fact a TCL procedure (script). Thus, Nuke also allows you to hand code generic TCL procedure to automate Nuke in various ways.

To source a generic TCL procedure:

1. Place the TCL procedure file in the plug-in path directory. Its name should include a `.tcl` extension.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. Create a menu option referencing the plug-in file (see "Defining Custom Menus and Toolbars" on page 622).

   Or instruct artists to invoke the TCL script by opening the Script Editor and entering nuke.load ("procedure/script file name") where procedure/script file name stands for the name of the procedure of script file without the extension.

Tip *For some code samples of useful Nuke TCL procedures, look inside the [Nuke directory]/plugins directory.*

Template Scripts

You can create a template script that is loaded instead of an empty script every time you launch Nuke or select **File** > **New** or **File** > **Close**. This allows you to save lookup table (LUT) setups and favorite arrangements of nodes, for example.

To create and use a template script:

1. Create the script you want to use as a template.
2. Select **File** > **Save as**. Navigate to ~/.nuke. The tilde (~) stands for your home directory and the full stop (.) for a hidden folder.
3. Name your script **template.nk** and click **Save**.

The next time you launch Nuke or select **File** > **New** or **File** > **Close**, Nuke loads the template from ~/.nuke/template.nk.

Tip *If you're not sure of the location of your home directory, on Linux and Mac you can open a terminal window and type **echo $HOME**. The terminal returns the pathname to your home directory.*

On Windows, you can find the .nuke directory under the directory pointed to by the HOME environment variable. If this variable is not set (which is common), the .nuke directory will be under the folder specified by the USERPROFILE environment variable. To find out if the HOME and USERPROFILE environment variables are set and where they are pointing at, enter %HOME% or %USERPROFILE% into the address bar in Windows Explorer. If the environment variable is set, the folder it's pointing at is opened. If it's not set, you will get an error.

Here are examples of what the pathname may be on different platforms:

***Linux**: /users/login name*

***Mac**: /Users/login name*

***Windows**: drive letter:\Documents and Settings\login name (XP) or drive letter:\Users\login name (Vista)*

Defining Common Preferences

The Nuke Preferences dialog (**Edit** > **Preferences**) allows any user to make myriad behavior and display adjustments to the interface. However, you may wish assign certain default preferences for artists.

To define default preferences:

1. Select **Edit** > **Preferences** to display the Preferences dialog.
2. Modify the controls within the dialog as necessary. For descriptions of what the controls do, see "The Available Preference Settings" on page 544.
3. Click **Save Prefs**. Nuke writes the modified preferences to a file called **`preferences6.nk`**, which is stored inside your [home directory]/**`.nuke`** directory.

Tip *If you're not sure of the location of your home directory, on Linux and Mac you can go to a terminal window and type **echo $HOME**. The terminal will return the pathname to your home directory.*

On Windows, you can find the .nuke directory under the directory pointed to by the HOME environment variable. If this variable is not set (which is common), the .nuke directory will be under the folder specified by the USERPROFILE environment variable. To find out if the HOME and USERPROFILE environment variables are set and where they are pointing at, enter %HOME% or %USERPROFILE% into the address bar in Windows

Explorer. If the environment variable is set, the folder it's pointing at is opened. If it's not set, you will get an error.

Here are examples of what the pathname may be on different platforms:

***Linux**: /users/login name*

***Mac**: /Users/login name*

***Windows**: drive letter:\Documents and Settings\login name (XP) or drive letter:\Users\login name (Vista)*

4. Move the resulting **`preferences6.nk`** file into your Nuke plug-in path directory.

   For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
5. If you haven't already done so, create a file called **`menu.py`** in your plug-in path directory.
6. Add the following entry in the **`menu.py`** file:

   ```
   nuke.load ("preferences6.nk")
   ```

Your preferences will now act as the defaults for your artists. However, should they make changes via Preferences dialog, these changes will override your defaults.

To delete (and reset) the preferences:

1. Open a terminal (or shell) as described for your operating system at the beginning of this chapter.
2. Using the prompt, go to the .nuke directory, under your home directory.
3. Enter **pwd** to display and verify the path.

   You should see something similar to

   - /users/*login name*/.nuke (on Linux),
   - /Users/*login name*/.nuke (on Mac) or
   - *drive letter*:\Documents and Settings\login name\\.nuke (on Windows XP) or

     drive letter:\Users\login name\\.nuke (on Windows Vista).

     This is not always the case, however, because on Windows the .nuke folder can be found under the directory pointed to by the HOME environment variable or (if HOME is not set) the USERPROFILE environment variable.

     To find out if the HOME and USERPROFILE environment variables are set and where they are pointing at, enter %HOME% or %USERPROFILE% into the address bar in Windows Explorer. If the environment variable is set, the folder it's pointing at is opened. If it's not set, you will get an error.

4. Enter `rm preferences6.nk` to delete the preference file.
5. Close the terminal or shell.

   The next time you launch Nuke, it will rebuild the file with the default preferences.

Altering a Script's Lookup Tables (LUTs)

Overview

A script's lookup tables are curves that control the conversion between file or device color spaces and Nuke's internal color space. In the curve editor, the x axis represents the input pixel values and the y axis the output pixel values (normalized to the 0-1 range). When applying LUTs, Nuke looks up the input value along the x axis to determine what the y value is to output.

Nuke provides the following set of default LUTs: **linear**, **sRGB**, **rec709**, **Cineon**[1], **Panalog**[2], **REDLog**[3], **ViperLog**[4], and **REDSpace**[5]. You can also create an unlimited number of additional LUTs and edit or remove existing LUTs in the script's settings.

By default, Nuke uses certain LUTs for certain file types or devices. In most cases, you do not need to touch these defaults. However, there may occasionally be cases when changing the defaults is necessary: for example, if your material has been shot with a camera that records in a custom color space, such as Panalog. In those cases, you can change the defaults in the script's settings so that you don't need to change the color space on each Read or Write node.

If you do not want to use the default LUT for reading or writing certain individual images, you can select the LUT to use in the corresponding Read or Write node's controls.

---

[1] The Cineon conversion is implemented as defined in Kodak's Cineon documentation.

[2] The Panalog LUT is based on a log2lin conversion with a blackpoint of 64, whitepoint of 681, and a gamma of 0.89.

[3] The REDLog LUT is based on a log2lin conversion with a blackpoint of 0, whitepoint of 1023, and a gamma of 1.022.

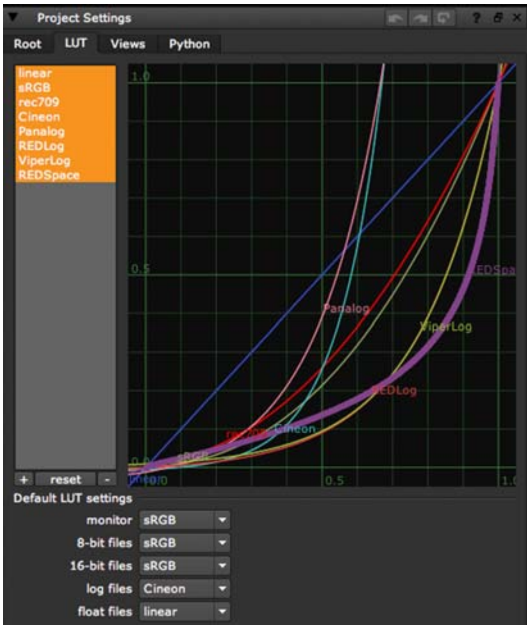
[4] The ViperLog LUT is based on a log2lin conversion with a blackpoint of 0, whitepoint of 1023, and a gamma of 1.0.

[5] The REDSpace LUT is implemented as defined in a curve provided by RED.

Displaying, Adding, Editing, and Deleting LUTs

To display LUT curves:

1. Select **Edit** > **Project settings** to open the settings for the script.
2. Go to the **LUT** tab.
3. From the list on the left, select the LUT you want to display in the curve editor. To select several LUTs, press **Ctrl** (Mac users press **Cmd**) while selecting the LUTs. All the selected LUTs are shown in the curve editor at the same time.


Project Settings
Root
LUT
Views
Python
linear
sRGB
rec709
Cineon
Panalog
REDLog
ViperLog
REDSpace
+
reset
-
1.0
0.5
0.5
1.0
Panalog
ViperLog
REDLog
Cineon
sRGB
REDSpa

Default LUT settings

monitor sRGB

8-bit files sRGB

16-bit files sRGB

log files Cineon

float files linear

To create a new LUT:

1. Select **Edit** > **Project settings** to open the settings for the script.
2. Go to the **LUT** tab.
3. Click the plus button (+). A dialog opens.
4. Enter a name for the new LUT and click **OK**.
5. Adjust the lookup curve to suit your needs. Click on the curve to select it. **Ctrl/Cmd**+**Alt**+**click** to add points on the curve, and drag the points to a new position. To change the shape of the curve, adjust the tangent handles.

The new LUT is now available in the global LUT settings, and the **colorspace** pulldown menu of Read and Write nodes' properties panels, the Viewer controls.

To edit LUTs:

1. Select **Edit** > **Project settings** to open the settings for the script.
2. Go to the **LUT** tab.
3. From the list on the left, select the LUT you want to edit.
4. Adjust the lookup curve to suit your needs. Click on the curve to select it. **Ctrl/Cmd+Alt+click** to add points on the curve, and drag the points to a new position. To change the shape of the curve, adjust the tangent handles.

   To use the usual editing commands, such as copy and paste, right-click on the curve editor and select **Edit**. Then, select the editing command you want to use, just like you would on any curve editor.

Note *Renaming existing LUTs is currently not possible. If you want to rename a LUT, you need to add and name a new LUT, copy the information from the old LUT into the new one, and then remove the old LUT.*

To reset the LUT curves back to their initial default shapes:

1. Select **Edit** > **Project settings** to open the settings for the script.
2. Go to the **LUT** tab.
3. From the list on the left, select the LUT you want to reset. To select several LUTs, press **Ctrl** (**Cmd** on a Mac) while selecting the LUTs.
4. Click **reset**.

To remove LUTs:

1. Select **Edit** > **Project settings** to open the settings for the script.
2. Go to the **LUT** tab.
3. From the list on the left, select the LUT you want to remove. Only remove LUTs that you have, for example, created by accident and are not using in your script. To remove the LUT, click the minus button (-).

The LUT is removed from the LUT settings, and the **colorspace** pulldown menu of Read and Write nodes' properties panels.

Note *If you remove a LUT that is used in a node, the node continues to refer to the LUT by name and raises an error.*

Selecting the LUT to Use

To select the LUT to use when reading or writing an image:

1. Double-click to open the Read or Write node's properties panel.
2. From the **colorspace** pulldown menu, select the LUT you want to use. To use the default LUT defined in Nuke's settings for the image type in question (see *Default LUT settings* below), select **default**.

Default LUT settings

By default, Nuke uses the following LUTs in the following cases:

File Type / Device	Default LUT
monitor. This is used for postage stamps, OpenGL textures, the color chooser display, and all other non-Viewer image displays. Also used by Truelight nodes when conversion to or from monitor color space is required.	sRGB
8-bit-files. This is used when reading or writing image files that contain 8-bit data. Also used by the Merge node's sRGB switch, and to convert Primatte and Truelight nodes' inputs into sRGB and their outputs from sRGB. The Cineon LUT is also used to control the input to Truelight.	sRGB
16-bit files. This is used when reading or writing image files that contain 16-bit integer data (not half float).	sRGB
log files. This is used when reading or writing .cin or .dpx files. Also used by Truelight nodes when conversion to or from log color space is required.	Cineon
float files. This is used when reading or writing image files that contain floating-point data.	linear

To change the Default LUT settings:

1. Select **Edit** > **Project settings** to open the settings for the script.
2. Go to the **LUT** tab.
3. From the pulldown menus under **Default LUT settings**, select the LUTs you want to use by default for each file type or device.

The new defaults are now used for any LUT setting where you have not selected a specific LUT. Any controls you have set to a specific LUT (that is, not set to **default**) will continue to use the selected LUT, and only those set to **default** will be affected.

Example Cases

Below are some examples of situations where you might need to alter the default LUTs.

Working in video color space

Emulating compositor software that works in video color space is not recommended. However, if you do need to do so, do the following:

1. Select **Edit** > **Project settings** and go to the **LUT** tab.
2. Under **Default LUT settings**, change the **monitor**, **8-bit files**, and **16-bit files** values to **linear**.

This prevents Nuke from converting from sRGB into linear. Nuke's nodes still assume linear data, but the image processing is applied to your unlinearized video color space images.

Linear data in 16-Bit files

Some facilities use linear data in 16-bit files. If this is the case in your facility, do the following:

1. Select **Edit** > **Project settings** and go to the **LUT** tab.
2. Under **Default LUT settings**, change the **16-bit files** value to **linear**.

Cineon displays

Some facilities have adjusted their monitor electronics to correctly display Cineon data. If this is the case in your facility, do the following:

1. Select **Edit** > **Project settings** and go to the **LUT** tab.
2. Under **Default LUT settings**, change the **monitor** value to **Cineon**.

Color management

Although true color management requires using the Truelight or other nodes, it may be useful to approximate it with a LUT that is used for the **monitor** setting. This way, texture maps and postage stamps resemble the final display more accurately.

If your color management is creating a monitor-corrected image, you'll want to set **monitor** to **sRGB** so you get reasonably monitor-correct output on non-Viewer images.

Creating Custom Viewer Processes

Using look-up tables (LUTs) in Viewer Processes, you can adjust individual Viewer displays to simulate the way the image will look on output to film or some video display device. Nuke includes some predefined Viewer Process gizmos, but you can also add your own processes by registering a node or gizmo as a Viewer Process. You can register as many custom Viewer Processes as you like. If you want to use one of the 1D LUTs listed in the Project Settings in your Viewer Process, you can use the built-in gizmo called ViewerProcess_1DLUT.

Tip *There are a couple of commented out examples in the installed init.py file demonstrating how to use a 3D LUT and a Truelight for a Viewer Process. You can find this file in the following location:*

On Windows:

drive letter:\Program Files\Nuke6.2v2\plugins or

drive letter:\Program Files (x86)\Nuke6.2v2\plugins

On Mac OS X:

/Applications/Nuke6.2v2/Nuke6.2v2.app/Contents/MacOS/plugins

On Linux:

/usr/local/Nuke6.2v2/plugins

All available Viewer Processes (both custom and predefined ones) can be applied from the Viewer Process menu in the Viewer controls.

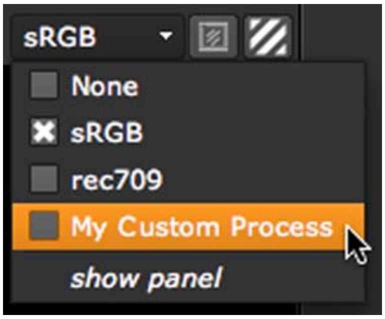


Figure 21.9: Both predefined and custom Viewer Processes can be applied from the Viewer Process menu.

Note that Viewer Processes are part of a built-in, fixed pipeline of nodes that are applied to images before they are displayed in the Viewer. This pipeline is either:

- gain > Input Process > Viewer Process > gamma > dither > channels > cliptest (if **viewer input order** has been set to **before viewer process** in the Viewer settings)

  OR
- gain > Viewer Process > Input Process > gamma > dither > channels > cliptest (if **viewer input order** has been set to **after viewer process** in the Viewer settings).

However, depending on what the Viewer Process is doing, this may not be the correct order. Therefore, if your Viewer Process (or an Input Process) has controls that also exist for the Viewer, such as controls named **gain**, **gamma**, or **cliptest**, then the Viewer will drive them from the corresponding Viewer controls and not do that image processing itself. This allows you to implement these controls in your Viewer Process using whatever nodes and order you want. If your Viewer Process does not have these controls (and they are not found on any Input Process in use either), then the Viewer will apply the effects in its normal way according to the built-in pipeline.

In the built-in pipeline, dither is applied to diffuse round-off errors in conversion of floating point data to the actual display bit depth. Although the cliptest is drawn at the end, it is computed on the image as input to the Viewer.

For more information on Input Processes, see "Input Process and Viewer Process controls" on page 103.

Using a Gizmo as a Custom Viewer Process

To create a custom Viewer Process, you would typically create a gizmo that includes some color correction like a look-up table (LUT) and register it as a Viewer Process using Python. (For more information on gizmos, see "Gizmos, Custom Plug-ins, and Generic TCL Scripts" on page 629.)

If you want to use one of the 1D LUTs listed in the Project Settings in your Viewer Process, you do not need to create a custom gizmo. Instead, you can simply register a built-in gizmo called ViewerProcess_1DLUT. This gizmo takes a parameter for which LUT to use, but does not allow it to be edited. For more information, see "To register a LUT in the Project Settings as a Viewer Process:" on page 653.

If you want anything more complex than a 1D LUT that can be found on the **LUT** tab of the Project Settings, you need to create your own gizmo and register that. For more information, see "To create a Viewer Process gizmo:" on page 654 and "To register a custom Viewer Process:" on page 655.

To register a LUT in the Project Settings as a Viewer Process:

1. Create a file called init.py in your plug-in path directory if one doesn't already exist. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. To register one of the LUTs in the Project Settings as a Viewer Process, use, for example, the following function in your init.py:

```
nuke.ViewerProcess.register("Cineon", nuke.createNode,
("ViewerProcess_1DLUT", "current Cineon"))
```

This registers a built-in gizmo called ViewerProcess_1DLUT as a Viewer Process and sets it to use the Cineon LUT. The registered Viewer Process appears in the Viewer Process menu as *Cineon*.

Note that you can set the built-in gizmo to use any 1D LUT in the Project Settings. For example, to set it to use the Panalog LUT, use the following function:

```
nuke.ViewerProcess.register("Panalog", nuke.createNode,
("ViewerProcess_1DLUT", "current Panalog"))
```

To create a Viewer Process gizmo:

1. Create the node(s) you want to use as a Viewer Process. For example, you can use a ColorLookup, Vectorfield (3D LUT), Truelight, or Colorspace node.
2. Select the node(s) you want to include in the Viewer Process and choose **Other** > **Group**.
3. To choose which controls the users of your Viewer Process can adjust, right-click on the dark gray background of the Group properties panel and select **Manage User Knobs**. For more information on how to add controls to your gizmo, see "Managing the gizmo controls" on page 631.

   If you expose controls with the same name as the controls in the Viewer (such as **gain** or **gamma**), then the controls in the Viewer will be used to drive these. However, if an Input Process that exposes the same controls is also in use, the Input Process will take precedence and the Viewer controls will drive it, ignoring the same-named Viewer Process control(s). For more information on Input Processes, see "Input Process and Viewer Process controls" on page 103.
4. Once you are happy with the modified Viewer Process group, export it to a gizmo by clicking **export as gizmo** on the **Node** tab of the group controls.
5. In the file browser that appears, click **Home**. Type **.nuke/** after the path displayed at the bottom of the file browser. Enter a name after the path, and append a **.gizmo** extension after the name. The name should begin with a capital letter. Finally, click **Save**.
6. Proceed to registering the gizmo as a custom Viewer Process, described below.

Tip *If you like, you can test your Viewer Process gizmo as an Input Process before registering it. Do the following:*

*1. In the top right corner of the Viewer, set the Viewer Process menu to **None**.*

2. Select the gizmo in the Node Graph.

*2. Select **Edit** > **Node** > **Use as Input Process**.*

3. To toggle the Input Process on or off, click the IP button in the Viewer controls. If you are happy with the result, proceed to registering the gizmo as a Viewer Process.

For more information on Input Processes, see "Input Process and Viewer Process controls" on page 103.

Tip *If you want to view or modify the internals of an existing Viewer Process, you can do the following:*

1. Select the Viewer Process that you want to modify from the Viewer Process menu.

*2. Select **Edit** > **Node** > **Copy Viewer Process to Node Graph**. This inserts the Viewer Process gizmo you selected in the Node Graph.*

*3. Double-click on the gizmo to open its controls. Go to the **Node** tab and click **copy to group**. This gives you an editable group version of the gizmo.*

*4. In the Group controls, click the **S** button to show the internals of the group. They are shown on a new tab in the Node Graph.*

*5. Make your changes and export the group to a gizmo by clicking **export as gizmo** on the **Node** tab of the group controls.*

Tip *If you use the ViewerLUT node in a Viewer Process gizmo, you can toggle **rbg_only** in the ViewerLUT controls to define whether the LUT is applied to all channels or only the red, green, and blue channels. You can also expose this control in the Viewer Process gizmo's controls, so that users can set it themselves.*

To register a custom Viewer Process:

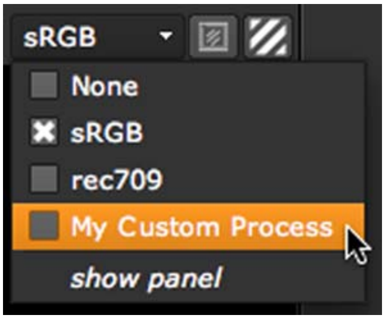
1. Create a file called init.py in your plug-in path directory if one doesn't already exist. For more information on plug-in path directories, see "Loading Gizmos, NDK Plug-ins, and TCL scripts" on page 616.
2. To register a gizmo or a node as a Viewer Process, use the following function in your init.py:

   nuke.ViewerProcess.register()

   For example, to register a gizmo called MyProcess.gizmo as a Viewer Process and have it appear in the Viewer Process menu as *My Custom Process*, you would enter the following:

   nuke.ViewerProcess.register("My Custom Process", nuke.Node, ("MyProcess", ""))

   Your Viewer Process should now appear in the Viewer controls.



If you need to unregister a Viewer Process, you can use **nuke.ViewerProcess.unregister()**. For example, **nuke.ViewerProcess.unregister("My Custom Process")**.

To get help on the use of these statements, you can enter **help (nuke.ViewerProcess)** in the Script Editor.

Tip *You can also pass arguments to nuke.ViewerProcess.register(). For example, to register a Blur node with its* ***size*** *knob set to 10, you would enter the following:*

nuke.ViewerProcess.register("Blur", nuke.createNode, ("Blur", "size 10"))

Tip *You can easily register any LUT defined in the Project settings as a Viewer Process. For how to do this, see the installed menu.py file where the built-in Viewer Processes are registered. You can find menu.py in the following location:*

On Windows*:*

drive letter:\Program Files\Nuke6.2v2\plugins or

drive letter:\Program Files (x86)\Nuke6.2v2\plugins

On Mac OS X:

/Applications/Nuke6.2v2/Nuke6.2v2.app/Contents/MacOS/plugins

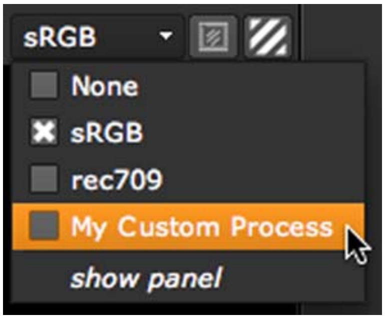
On Linux:

/usr/local/Nuke6.2v2/plugins

Applying Custom Viewer Processes to Images

To apply your custom Viewer Process to images displayed in a Viewer:

Select the process from the Viewer Process pulldown menu in the Viewer controls.

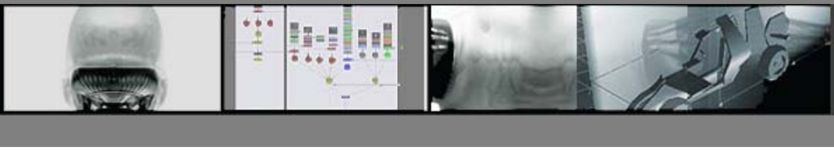


To view the controls of the currently active Viewer Process:

In the Viewer controls, select ***show panel*** from the Viewer Process menu.

This opens the Viewer Process' properties panel. Any controls with the same name as the controls in the Viewer (such as **gain** or **gamma**) can only be adjusted using the Viewer controls. If these controls are also exposed on an Input Process and the Input Process has been activated, the Viewer controls drive the Input Process controls and the Viewer Process controls are disabled.

For more information on Input Processes, see "Input Process and Viewer Process controls" on page 103.



NUKEX

You may wonder what the main differences are between the familiar Nuke and NukeX. In the following pages, you're going to learn all about what's different between the two, how to install and license NukeX, and more.

NukeX Features

When using NukeX, you have all the features of Nuke in use, plus the following:

- **CameraTracker** - With the fully integrated 3D CameraTracker node, you can do you own camera solves and create reference geometry and cards positioned at tracked points in the 3D scene. For more information, see "Camera Tracking" on page 661.
- **PointCloudGenerator** - You can create dense point clouds from your footage using the PointCloudGenerator and CameraTracker. For more information, see "Creating Dense Point Clouds and 3D Meshes" on page 687.
- **PoissonMesh** - With the PoissonMesh node, you can use a dense point cloud to create a 3D mesh from your 2D footage. For more information, see "Creating a Mesh Using a Point Cloud" on page 689.
- **Lens Distortion** - The LensDistortion node gives you multiple ways to analyze image sequences and lens grids, resulting in a lens model and the ability to warp and un-warp in order to compensate for lens distortion. For more information, see "Adding and Removing Lens Distortion" on page 680.
- **DepthGenerator** - The DepthGenerator node provides a method to produce a per-frame Z-depth map from the input 2D footage. It additionally requires a camera solve which can be obtained using the CameraTracker node. For more information, see "Generating Depth Maps" on page 692.
- **Modeler** - You can create polygon faces from your 2D footage and use them to make a 3D model with the Modeler node. You can create both faces and single vertices and view them in the 3D view, and you can edit and extrude the faces after creating them. For more information, see "Using the Modeler Node" on page 696.
- **ProjectionSolver -** You can now align 2D images with a 3D model using the ProjectionSolver node. You can match points in your 2D footage with corresponding points in 3D, and the ProjectionSolver node will use this information to create a solved camera to match your footage. You can also create Card nodes to correspond with your camera positions to help

you visualize your scene better. For more information, see "Using ProjectionSolver" on page 701.
- **PrmanRender** - PrmanRender is a new render node in NukeX that works together with Pixar's PhotoRealistic RenderMan® software to give you an even better quality render result. PrmanRender is an alternative to the ScanlineRender node for rendering 3D scenes, and it gives you control over features such as shadows, reflections, refractions and depth-of-field. For more information, see "Rendering with PRmanRender" on page 706.
- **FurnaceCore** - This plug-in bundle consists of twelve of The Foundry's best Furnace tools including Kronos, the optical flow re-timer, grain and de-grain tools, rig-removal, and more. For more information, select **Help** > **Documentation** in Nuke and have a look at the FurnaceCore User Guide.

Nuke and NukeX are fully script compatible, with Nuke capable of viewing and rendering - but not editing - NukeX nodes.

Installing NukeX

NukeX gets installed with Nuke 6.0 (and later). If you already have Nuke 6.0 (or later) installed and have obtained a license for both Nuke and NukeX, you automatically have access to the NukeX features, including FurnaceCore. During the Nuke installation, three shortcuts will be created for you: one for Nuke, one for Nuke Personal Learning Edition (PLE), and one for NukeX. For more information on the details of the installation process, see "Installation and Licensing" on page 25.

Licensing NukeX

To use NukeX, you need both a Nuke and a NukeX license. The NukeX license won't replace your Nuke license, which ensures you can still run previous versions of Nuke. For more information on licensing, see "Installation and Licensing" on page 25.

Launching NukeX

To run NukeX, follow the below instructions.

On Windows

Do one of the following:
- Double-click the **NukeX** icon on the Desktop.
- Select **Start** > **All Programs** > **The Foundry > Nuke**6.2v2 (32/64 bit) > **NukeX**6.2v2 (32/64 bit).

- Using a command prompt, navigate to the Nuke application directory (by default, \Program Files\Nuke6.2v2), and enter **Nuke6.2v2 --nukex**.

On Linux

Do one of the following:

- Double-click the NukeX icon on the Desktop.
- Click on the NukeX Quick Launch icon created in the Applications list.
- Open a terminal, navigate to the Nuke application directory (by default, /usr/local/Nuke), and enter **./Nuke6.2v2 --nukex**.

On Mac OS X

Do one of the following:

- Click the NukeX dock icon.
- Open the Nuke application directory (by default, **/Applications/Nuke6.2v2/**), and double-click the **NukeX** icon (or list item).
- Open a terminal, navigate to /Applications/Nuke6.2v2-32/Nuke6.2v2.app/Contents/MacOS, and enter ./Nuke6.2v2 --nukex.

For more information on using the command line, see "What Is a Terminal and How Do I Use One?" on page 603.

Note *On Mac OS X, you shouldn't move the NukeX bundle from the folder where the Nuke bundle is located as this will prevent NukeX from working correctly.*

1 CAMERA TRACKING

Nuke's CameraTracker node is designed to provide an integrated camera tracking or matchmoving tool which allows you to create a virtual camera whose movement matches that of your original camera. Tracking camera movement in a 2D footage enables you to add virtual 3D objects to your 2D footage.

With the CameraTracker node, you can track the motion in the input 2D footage to create a 3D camera. You can automatically track features or add manual tracks from a Tracker node, mask out moving objects using a Bezier or B-spline shape and edit your tracks manually. You can also solve the position of several types of cameras as well as solve stereo sequences. CameraTracker will automatically create a scene linked to the solve containing a 3D camera and point cloud.

Quick Start

The process of tracking camera movement and creating a virtual camera is roughly the following:

1. Connect the CameraTracker node to the sequence you want to track. See "Connecting the CameraTracker Node" on page 662.
2. If you want, you can seed points in your footage. This means choosing specific points you want to track alongside other features that will be tracked automatically. For more information, see "Seeding Tracks" on page 662.
3. Set the tracking parameters. Click **Track Features** to track the sequence. These are described under "Setting Tracking Parameters" on page 662.
4. Set the camera parameters. These are described under "Adjusting the Camera Parameters" on page 667.
5. To account for lens distortion in your footage, adjust the controls on the **Lens** tab to calculate it and use it in the camera solve. For more information, see "Accounting for Lens Distortion" on page 668.
6. Solve the Camera position by clicking **Solve Camera** and refine it, if necessary, on the **Refine** tab. For more information, see "Adjusting the Solve" on page 670. See also "Troubleshooting the Solve" on page 675.
7. Create a 3D scene, a point cloud, and a camera by clicking **Create Scene**.
8. Adjust the resulting scene with the **Scene Transform** parameters on the **Scene** tab. See "Transforming the Scene" on page 674.
9. Add your 3D virtual objects to the footage. See "Attaching Objects to the Footage" on page 677.

Connecting the CameraTracker Node

1. Load and select the clip you want to track.
2. Click **3D** > **CameraTracker.**
3. If you want to omit a part of the scene from being tracked, connect a matte to the **Mask** input. Note that, unlike the **source** input, this input is hidden and appears as a small triangle on the left hand side of the node. For more information about masking, see "Masking Out Regions of the Image" on page 664.
4. Click **Image** > **Viewer** to insert a Viewer node and connect it to the CameraTracker node.

Tracking Features in a Sequence

The CameraTracker node tracks footage attached to the **Source** input on the node. CameraTracker then analyzes the input in two stages. Tracking defines the set of 2D feature tracks that correspond to fixed rigid points in the scene. The solver then calculates the camera path and projection that creates a 3D point for each feature track with a minimum projection error.

Seeding Tracks

Before you start tracking your footage, you can choose specific points in your sequence which you know you'll want tracked. This is called seeding and you can do it as follows:

1. Right click on the point in your footage you want to seed
2. Select **Tracks > Seed track**.

   An orange cross indicates the location of your track, and when you click **Track Features**, your seeded point will be tracked with the other automatically chosen features.

Setting Tracking Parameters

First, you'll need to adjust a set of tracking controls in order to get the best possible tracking results.

1. On the **CameraTracker** tab, define which parts of your footage CameraTracker should track.
   - With **Analysis range,** you can set a specific range of frames to track and solve. Choose **Source Clip Range** to track and solve the entire sequence or **Analysis Range** to define a frame range. Enter the first and last frame of your range to **Analysis Start** and **Analysis Stop** boxes respectively.
   - If you want to mask out parts of your image, set **mask** to anything other than **None** (the default). For more information, see "Masking Out Regions of the Image" on page 664.
2. On the **Tracking** tab you can further define settings for feature tracking.

- **Number of Features** - Define the number of features you want to track in each frame. The default is 150 features and ideally you should use more than 100 tracks per frame. In difficult sequences to solve, consider using higher number of features.
- **Detection Threshold** - Set the distribution of features over the input image. If you enter a low detection threshold value, features will be tracked evenly on all parts of the image. Use **Preview Features** to preview your features before tracking.
- **Feature Separation** - Set the distribution of features in relation to each other. To force feature separation and spread features evenly over the image at even distances, enter a high feature separation value. Use **Preview Features** to preview your feature separation before tracking.
- **Refine Feature Locations** - Check this to lock detected features to local corners. If you activate this, CameraTracker will find the closest corner point in your footage and lock the feature point to it.
- **Preview Features** - Check this box to preview the features that will be tracked. Preview comes in handy when you want to tweak the tracking parameters further before tracking.

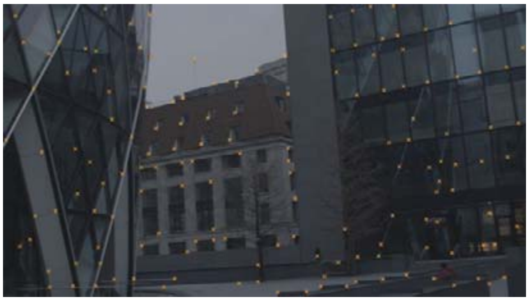


Figure 1.1: Previewing features on your footage.

- **Minimum Length -** Set a threshold value for the minimum track length to reject short tracks. You might find a lot of short tracks cropping up in long sequences with a slow camera movement.
- **Track Threshold** - Set a threshold value between 0 and 1. This threshold controls how similar features look over a number of frames. You can adjust this value to test whether a track is reliable.
- **Track Smoothness** - Set the threshold for smooth track generation. Adjusting this value can be useful in preventing poor tracks in complex sequences. Increase the smoothness value to remove tracks that glitch over time.
- **Track Consistency** - Set the threshold for consistent track generation. Increase this value to ensure track motion is locally consistent. Adjust consistency to prevent poor tracks in complex sequences.

- **Track Validation** - Tell CameraTracker what type of camera movement the tracks should fit. Use **free camera** and **rotating camera** options to test tracks against the expected motion of the camera. Select:
    - **Free camera** - Expect a freely moving camera that is rotating and translating.
    - **Rotating Camera** - Expect a still camera that is rotating only.
    - **None** - Do not validate tracks based on any particular camera movement. You might want to use this validation option when you need a larger number of tracks and want to keep all tracks in the image.

3. When you're happy with your **input** settings, click **Track Features**.

Masking Out Regions of the Image

If you don't want to track all regions of your image, for example if there are moving objects in the image, you can attach a matte to the **Mask** input to define image regions that should not be tracked. You can also use the source input's alpha channel as a matte.

1. If you want to use a separate matte for masking, connect it to the **Mask** input in the Node Graph.
2. In the Properties panel, set **Mask** to the component you want to use as a mask:
    - **None** - Track features in the whole footage.
    - **Source Alpha** - Use the alpha channel of the source clip to define which areas to ignore.
    - **Source Inverted Alpha** - Use the inverted alpha channel of the source clip to define which areas to ignore.
    - **Mask Luminance** - Use the luminance of the mask input to define which areas to ignore.
    - **Mask Inverted Luminance** - Use the inverted luminance of the mask input to define which areas to ignore.
    - **Mask Alpha** - Use the mask input alpha channel to define which areas to ignore.
    - **Mask Inverted Alpha** - Use the inverted mask input alpha channel to define which areas to ignore.

Viewing Tracks and Track Information

You can also preview your tracks and track information. On the **CameraTracker** tab, next to **Display**, select:

- **2D Point** - Check to view a circle around the reprojected points in the 2D Viewer, and the error between the tracks and the points.

- **3D Marker** - Check to view cone shaped 3D markers on your points in the 3D Viewer. You can set the size of the markers using the **Marker Scale** parameter.
- **Key Tracks** - Check to view only the feature tracks, which are visible in your key frames. This is useful if you want to reduce the size of your point cloud and display the most reliable 3D points.

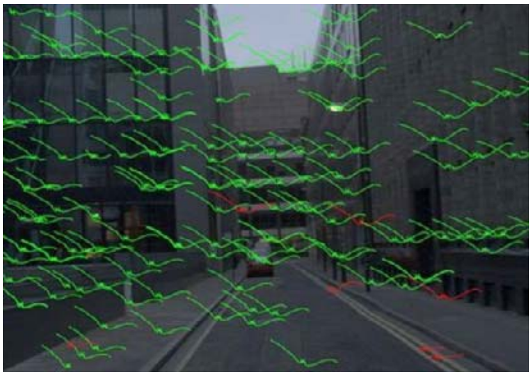


Figure 1.2: View all tracks.

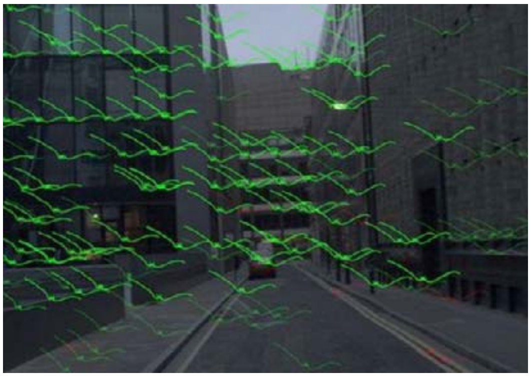


Figure 1.3: View key tracks only.

Creating Manual Tracks

CameraTracker node is mainly designed for automatic tracking. After automatic tracking, you can add user tracks on particular scene points either to extract a specific set of points or to lock the solve onto a particular part of the scene. You can also convert automatically tracked points into user tracks. If you check the **import/export auto-tracks** box on the **Tracking** tab, you can read and write the automatic tracks as well as the user tracks with other nodes.

If you want, you can use manual tracks only for tracking your footage. In this case, automatic tracking is not necessary and you only need to set your user tracks on the **Tracking** tab and move directly to solving the camera (see "Solving the Camera Position" on page 669). However, when using manual tracks only, you need to make sure there are sufficient tracks available to solve the cameras. Depending on how many key frames you want to use for initializing the solve, you need to have at least six or seven tracks (see "Solving the Camera Position" on page 669), but ideally more tracks are required to get a good solve result.

To import tracks from the Tracker node

1. In the CameraTracker node controls, on the **Tracking** tab, click **Add User Track** as many times as you need to create the number of tracks you want to add.
2. From the Tracker node control panel, drag your track values on one of the user track rows you created for CameraTracker.
3. Click **Solve Camera** and **Create Scene** to solve the scene and create a point cloud with your user tracks only.

   You can also click **Track Features** after creating your user tracks. In this case, Camera Tracker will track your user tracks as well as creating automatic tracks.

To import user tracks from a tracks file

1. On the **Tracking** tab, click **Import User Tracks**.
2. In the **Import File** dialog, browse to find the tracks file you want to import from. Your user track file doesn't need to have any specific extension or format.
3. To import track values, click **Import User Tracks** and track values will be read from your tracks file.

To convert an automatically tracked point into a user track

You can also take an automatically tracked point and turn it into a user track.

1. Click **Track Features** on the **CameraTracker** tab. This starts the automatic tracking on your sequence (or a part of it). See "Tracking Features in a Sequence" on page 662 for further instructions on this.
2. Select a tracked feature, right-click it and select **tracks > extract user track**.

   The track you selected is circled in the Viewer and now appears as a user track on the **Tracking** tab, under **User Tracks**.

To export and delete tracks

1. On the **Tracking** tab, click **Export User Tracks**.
2. In the **Export File** dialog, browse to find the tracks file you want to export to. All your user tracks will be written into the tracks file.
3. To delete user tracks, click **Delete** next to your track.

Setting the Camera Parameters

Once you've set your tracking parameters and created a set of feature tracks on your image, you can move on to adjusting parameters for your camera solve.

Adjusting the Camera Parameters

You can adjust the following camera solve settings on the **Solver** tab in the properties panel.

- **Focal Length Type** - Set the focal length for the camera.
  - **Unknown Constant** - Select this if there's no zoom and the focal length is unknown.
  - **Unknown Varying** - Select this if the focal length is unknown and changes.
  - **Approximate Constant** - Select this if there is no zoom and an approximate focal length is available and enter the focal length value next to **Focal Length**.
  - **Approximate Varying** - Select this if an approximate focal length is available and changes and enter the focal length value next to **Focal Length**.
  - **Known** - Select this if the focal length is defined for the sequence and enter the focal length value next to **Focal Length**.
- **Focal Length** - Set the focal length for approximate and known solves. You can animate focal length to define a varying focal length. The units should match the units used for the film back size (millimeters, for example).
- **Film Back Size** - Set the size of the imaging sensor of the camera and specify the units you want to use by selecting either **millimeters** or **inches** in the **Units** drop-down. You only need to specify this if you have also specified focal length. The units should match the units used for the focal length.
- **Camera Motion** - Set the type of motion for the camera. Select:
  - **Rotation Only** - Select this if the camera is still and only rotating.
  - **Free Camera** - Select this if the camera is moving freely, rotating and translating.

  - **Linear Motion** - Select this if the camera has a straight, linear path.
  - **Planar Motion** - Select this if the camera has a flat path, moving in a two-dimensional plane only.
- **Smoothness** - Adjust this to smooth your camera path. Increase the value to add weighting to the camera path and to create a smoother result.

Accounting for Lens Distortion

You can use CameraTracker's lens distortion feature to account for lens distortion in your footage. You can then track and solve from the original footage and calculate the camera for the undistorted footage. On the **Lens** tab, you can define the distortion type in your footage and adjust other lens distortion controls:

1. In the Lens Distortion drop-down, select the type of distortion to expect.
   - **No Lens Distortion** disables all lens distortion controls and treats the footage as having no distortion. This option gives the best solve to work on the original footage. The calculated focal length will compensate for lens distortion.
   - **Known Lens** allows you to specify the lens distortion manually and to use that information in doing the camera solve. Use this when you have a grid available for defining lens distortion. You can calculate the distortion parameters in a LensDistortion node first and paste the values onto the CameraTracker node.
   - **Refine Known Lens** gives you a chance to give an approximate distortion which to use but attempts to refine it in the camera solve. This is not necessary when a grid is used but could improve the solve if lens distortion is calculated using an alternative method or set manually.
   - **Unknown Lens** calculates the lens distortion automatically from the se quence (in the same way as the Image Analysis option in the LensDistortion node) and then refines the distortion in the camera solve. This way you don't have to calculate your lens distortion separately using the LensDistortion node.
2. If you chose either **Known Lens** or **Refine Known Lens**, enter values for the **Lens Type**, **Radial Distortion**, **Distortion Center** controls and, if necessary, the **Anamorphic Squeeze**, **Asymmetric Distortion** and **filter** controls. For more information on these controls, see "Adjusting LensDistortion Parameters" on page 683.
3. Move straight on to clicking **Solve Camera** and **Create Scene** to view your results or further adjust the solver controls on the **Solver** tab.

   After you have your lens distortion values, you can also toggle the **Undistort Input** box to view your footage distorted or undistorted.

Using lens distortion in a Card node

CameraTracker automatically recalculates the lens distortion parameters so you can choose to apply the lens distortion model in a Card node. Do the following:

1. Open the **Card Parameters** folder on the **Lens** tab.
2. Hold down **Ctrl/Cmd** and drag the **x**, **y** and **z** values from the **scale** parameter's values on to the corresponding fields on the **Card** tab in the Card node (**3D** > **Geometry** > **Card**) control panel. This will set the right scale for the reproduced distortion on the Card.
3. Hold down **Ctrl/Cmd** and drag the **a**, **b,** and **c** values onto the corresponding fields on the **Lens Distortion** tab in the Card node control panel. These values are now used to undistort the image attached to the **img** input of the Card node.

Solving the Camera Position

When you're happy with the features that you've tracked, you can proceed to solve the camera position. CameraTracker uses the tracking information to calculate the camera position and add position information to the feature points in the Viewer.

The solver first automatically selects a subset of frames called keyframes that describe the camera motion. Key frames are usually positioned relatively far apart so that there is sufficient parallax in the images to define the 3D points for the tracks. This enables a better definition of the 3D points and makes the solve more accurate. The key frames are solved first then the rest of the sequence, so keyframe selection can be critical in solving complex footage.

If you find the solve isn't very good, you can try changing the keyframe parameters for the solver.

1. If necessary, adjust the **Keyframes** parameters on the **Solver** tab.
   - **Keyframe Separation** - Use a high separation to spread key frames out in long sequences with small camera movements. Use a low separation to generate more key frames for fast camera moves.
   - **Reference Frame** - Set the first frame to use as a keyframe in the solve. This should be a frame where there is a large number of tracks distributed over the image with a variation in depth.
     - **Set reference frame** - Check here to enable **Reference Frame** field and manually define how key frames are specified in the sequence. Choosing to do this can be useful if you have a difficult sequence to solve.

- **Keyframe Accuracy** - Choose **Low**, **Medium** or **High** to adjust how accurate your keyframes are. A higher accuracy will take a longer time to solve but it can improve the result when you're working with a long sequence.
    - **Three Frame Initialization** - Switch to using three key frames rather than two to initialize the solve.

2. If you need to, you can adjust the camera output values. See "Adjusting the Virtual Camera" on page 674.
3. To view your results, click **Solve Camera**, and move on to either adjusting your solve or create a scene straight away by clicking **Create Scene**.
4. You can also check the **link output** box to create a link between CameraTracker and the nodes that CameraTracker outputs, so that when the solve updates, the Cameras and points are updated too.

Tip *Keyframes allow the solver to use a subset of frames where the camera positions are relatively widely spaced. This ensures that 3D positions can be defined more reliably (in triangulation). When solving a sequence from a digital camera, the distance between camera positions can be relatively large already, so try reducing the* ***Keyframe Separation*** *to generate more keyframes.*

Clearing Automatic Tracks

Sometimes you may want to use CameraTracker for just creating a few user tracks and a solved camera. For example, you might want to create a solved camera and use with the Modeler node to create 3D faces from your 2D footage. In this case, you can get rid of the automatically created tracks and 3D points at this stage by clicking **Delete Auto-tracks** in the CameraTracker node control panel.

Adjusting the Solve

After you've tracked and solved your footage, you can evaluate your tracks, delete any tracked features you do not want to keep, and use the **Refine** tab features to refine your solve further. If you're having problems with your solve, also have a look at "Troubleshooting the Solve" on page 675.

Deleting Tracks

You can delete features that have been tracked. You might want to do this when moving objects or features that don't correspond to true 3D points (such as reflections) have been tracked.

To delete tracked features

1. In the Viewer, select the track you want to remove. You can also marquee select points and select several tracks by holding down **Shift** while selecting points.
2. Right-click on the selected track(s) and select **Tracks > Delete selected**.

   If you've already solved the camera, created a scene and removed and edited points in it, you can just click **Solve Camera** again, and the scene will automatically update.

Tip *Sometimes sequences generate very long tracks. Long tracks are good, as this indicates a reliable 3D point. However, if tracks are too long, the pixels that are tracked in the image can end up drifting from the underlying 3D point. When this happens, the track will be rejected in the solve as the error between the 3D point and the 2D track becomes large. To stop tracks getting too long, just increase the* ***Track Threshold*** *value.*

Refining the Solve

On the **Refine** tab, the first section of the tab, **Track info**, gives you detailed information of the solve and in the Analysis section you can redo your solve, recalculate it and adjust your output.

The **Solve Error** field at the top of the tab gives you the final RMS (root mean square) error of your solve. The **Keyframes** control displays the automatically selected keyframes for the solve.

Under **Track Info**, in the **Track Display** drop-down, select:

- **Tracks** - Select to view track status, which is visible as amber if the track is unsolved, green if it's solved and red if it's invalid or rejected. Point to a single track to view **Track Length**.

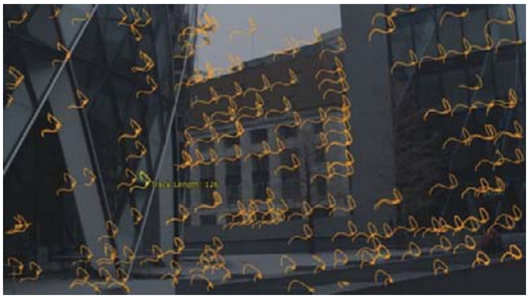


Figure 1.4: Viewing tracks only,

- **Point RMSE** - Select to evaluate the reliability of your tracks by viewing the root mean square reprojection error of your 3D points. This is the error between the calculated 3D point for a track and the tracked 2D feature in the 2D image. It is the mean error across the lifetime of each track. The tracks are color coded red (for an unreliable point), yellow (for a potentially unreliable point) or green (for a reliable point). Point to a single track to view its **Track Length** and **Reprojection Error** information.

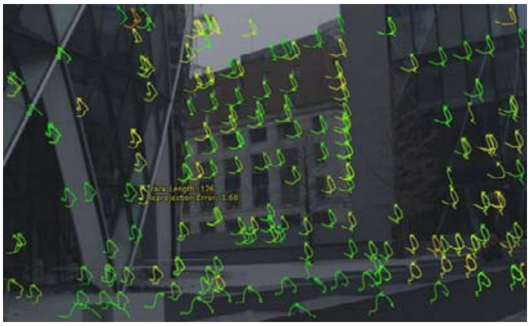


Figure 1.5: Point RMSE indicated with colors.

- **Point Error -** set to view the reprojection error of your 3D points. This is the error between the calculated 3D point for a track and the 2D feature for a track at the current frame.The tracks are color coded red, yellow or green according to their reprojection error figures. Point to a single track to view its **Track Length** and **Reprojection Error** information.

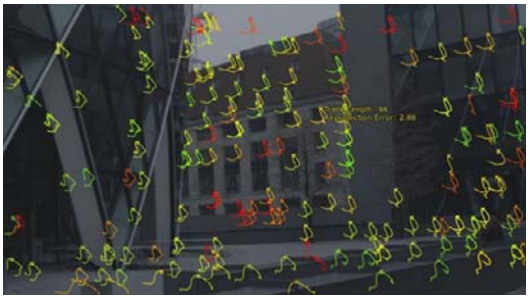


Figure 1.6: Viewing point error.

In the **Track Curves** graph, you can view the statistical details of your track and solve as curves. The curves show length and error information on your tracks. They are set automatically, and you should not edit them manually in the graph. Instead, use the **Analysis** section of the tab to make changes, such as exclude tracks.

The curves in the Track Curves graph show the following details:

- **num tracks** - the number of tracked features at each frame.
- **track len - min** - the minimum length of the tracks at each frame (in frames).

- **track len - avg** - the average length of the tracks at each frame (in frames).
- **track len - max** - the maximum length of the tracks at each frame (in frames).
- **Min Length** - the threshold for minimum track length.
- **Solve Error** - displays the constant **Solve Error** parameter.
- **error min** - the minimum reprojection error at each frame (in pixels).
- **error rms -** the root mean reprojection error at each frame (in pixels).
- **error track -** the maximum root mean reprojection error calculated over the track lifetime at each frame (in pixels)
- **error max -** the maximum reprojection error at each frame (in pixels).
- **Max Track Error -** displays the constant **Max RMS Error** parameter.
- **Max Error -** displays the **Max Error** threshold parameter.

Under the **Track Curves** view, you can adjust your thresholds and adjust your solve if you find your solve or camera output needs readjusting.

- **Min Length** - Increase the minimum length threshold to reject short tracks. You might find a lot of short tracks cropping up in long sequences with a slow camera movement.
- **Max Track Error** - Reduce this threshold to reject tracks based on RMS reprojection error.
- **Max Error** - Reduce this threshold to reject tracks with a large reprojection error in isolated frames.

In the **Analysis** section, you can adjust your solve in different ways and delete unwanted tracks:

- **Solve Camera** - Recalculate your solve in the same way as on the **CameraTracker** tab. A new solve is calculated irrespective of the threshold that you've set on the **Refine** tab. You may want to resolve from scratch after deleting tracks using **Delete Unsolved** or **Delete Rejected**.
- **Recalculate Solve** - Recalculate your solve from the inlier tracks that remain after you've adjusted your threshold values. Recalculating the solve is handy when you want to see how good the solve is after you've refined it, without permanently deleting tracks. Recalculating the solve also recalculates the stereo geometry.
- **Refine Output** - This starts with the current camera solve and uses the inlier tracks (that is the tracks that have not been rejected by the thresholds) to fine tune focal length, camera position or camera rotation (or a combination of these). You can manually edit the camera solve first on the **Ouput** tab, then choose:
  - **Focal Length** - Check to refine the camera's focal length.

  - **Position** - Check to refine the camera position.
  - **Rotation** - Check to refine the camera rotation.

  For example, you might want to smooth the camera path then use **Refine Output** to update the camera rotation to match the camera path.
- **Delete Unsolved** - Click to permanently delete tracks for which 3D points could not be calculated in the solve.
- **Delete Rejected** - Click to permanently delete tracks that have been rejected by the thresholds. For more information on deleting tracks, see "Deleting Tracks" on page 670.

Transforming the Scene

After you've created your scene, you can use the transform controls on the **Scene** tab to fine-tune the result. All the settings on this tab are animatable.

- **rotation order** - Set the operation order for rotations from the dropdown menu which displays all the possible axial combinations.
- **translate** - Set translate values for the scene. You can adjust translate values by clicking and dragging the axis in the 3D viewer.
- **rotate** - Set values to rotate your camera or use **Ctrl/Cmd** to rotate the camera in the Viewer.
- **uniform scale** - Set a scale for your scene. For more information about scaling, see "Scaling a Scene" on page 677. Alternatively, you can also use the **Local matrix** and **World matrix** to perform transforms on your scene.

Adjusting the Virtual Camera

On the **Output** tab, you can adjust the controls for the virtual camera created by the CameraTracker node. In most circumstances, you shouldn't need to touch these at all as they provide additional ways to setup the virtual camera which CameraTracker automatically creates.

- **Translate** - Define transform values for the virtual camera's position.
- **Rotate** - Define transform values for the virtual camera's rotation.
- **Focal Length** - Set the focal length for the virtual camera.
- **Aperture** - Set the aperture angle in degrees for the virtual camera.
- **Window Translate** - Set the center point offset for the camera projection.
- **Window Scale** - Set the relative pixel scaling value for the camera projection.

Tip *You might want to use the virtual camera output settings in a case, where you want to smooth a noisy solve. You can use the Curve Editor to see your* *values and right-click **Interpolation** > **Smooth** to smooth them. You can also smooth the focalLength result on the Output tab then copy over to the **Solver** tab to perform a Known focal length solve. This way you're guaranteed to get a smooth change in focal length in your solve.*

Centering on Selected Tracks

You can center the Viewer to a particular track or several tracks. This way you can see how well your tracks stick to the image. To center the Viewer:

1. Select one or more tracks in the Viewer
2. Right-click on your tracks and select **tracks > center selected**.
3. The Viewer now centers the selected tracks while you view your footage. You can turn centering off by right-clicking anywhere in the Viewer and selecting **center viewer off**.

Troubleshooting the Solve

If you're having problems getting the results you want with your solve, there are a few things you can try:

- If you have a tricky sequence which simply doesn't seem to solve, your best course of action might be to reduce the **keyframe separation** value on the **Solver** tab to 0.1, and then try solving for different reference frames. You can also try using **Three frame initialisation** and a high keyframe accuracy value, but note that this process is rather slow.
- If your camera does solve but drifts away from where it should be, then using long user tracks could help you to lock down the camera. You can add and import user tracks on the **Tracking** tab. You could also try putting the **keyframe separation** value at 0.1 and using high keyframe accuracy to stop the drift.
- Solving long, slow sequences can introduce camera jitter. In this case you might try setting the **camera smoothness** on the **Solver** tab to 0.1 to smooth out the jitter on the camera path.

Using the Point Cloud

At this point, you should have a scene, a camera, and a point cloud created by the CameraTracker node and linked to the result of the solve. You can use the point cloud to position scene elements.

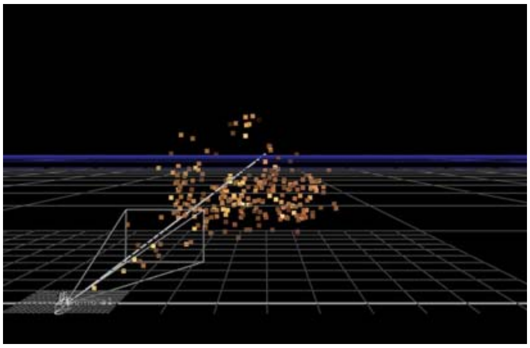


Figure 1.7: Viewing the point cloud in 3D view.

Tip *If your point cloud is very dense, you can reduce the size of it by only viewing the tracks visible in your key frames. That way you can view less and more accurate 3D points. To reduce the point cloud size, check the **Key Tracks** box on the **CameraTracker** tab.*

Setting Points on Ground Origin or Different Axes

After you've created your scene you can select points on your footage and tell CameraTracker to set them on a particular axis (Y, X or Z). Do the following:

1. Select points (more than one) on your footage that you want to mark as belonging to a particular axis.
2. Right-click on one of them and select **ground plane** > **set X**, **set Y** or **set Z** depending on which axis you want to set the points on.
3. You can also set ground origin, or the center point of the X, Y and Z axes, by right-clicking on a point and selecting **ground plane > set origin**.

Note *This can only be done in 2D. You can switch to 3D view to check the validity of points before setting the axes or origin.*

Setting a Ground Plane

You can mark features in the footage as belonging to the ground level of the footage and thus define the ground plane. The scene transform is applied after the ground plane is set so you may want to set the ground plane first before defining the transform in the **Scene** tab.

1. Select a frame with a large number of tracks covering the ground region.
2. Select the track you want to mark as ground plane. You can also marquee select tracks and select several tracks by holding down **Shift** while selecting points.
3. Right-click and select **ground plane** > **set to selected.**

4. If you want, you can move to a different frame and add further points to the ground plane.
5. You can also reset the ground plane if you're not happy with it. To do so, right-click on a ground plane point and selecting **ground plane** > **reset**.

Note *You can only set the ground plane in 2D. You can switch to 3D view to check the validity of points before setting the axes or origin.*

Scaling a Scene

You can define the scale of the scene with the **uniform scale** and **Scale Constraints** parameters on the **Scene** tab. Note that you can only scale the scene this way in the 2D view.

1. Set your scene scale with the **uniform scale** slider.

   If you need to define a more specific scale, you can add **Scale Constraints** to define your scale manually. Do the following:
2. Select two (and only two) tracks using **Shift**+click in the Viewer overlay.
3. Right-click on one of the tracks and select **scene** > **add scale distance**.
4. In the properties panel add the distance between the two points to the first **Distance** row under **Scale Constraints**. The scale of your scene is adjusted according to the distance you've entered.
5. You can also delete constraints if, for instance, you find that some of them are not as accurate as your other constraints.

Attaching Objects to the Footage

Once your scene is set, you can start attaching 3D objects to the point locations.

1. Select a track or a set of tracks where you want to attach your object.

   Selected tracks are highlighted in yellow. You can attach another Viewer node to the CameraTracker node to see the highlighted 3D points alongside the 2D view. This way you can ensure the points are where you expect them to be.
2. Right-click on your selection and choose an object you want to attach to your footage:
   - Select **create > cube**, **cylinder**, **sphere** or **card** to create a basic shape.
   - Select **create** > **card XY**, **card YZ** or **card ZX** to create a card which is aligned with the x, y or z axis.
   - Select **create** > **axis** to create a new axis.

Copying Translate and Rotate Values

You can copy the translate and rotate values directly from your points and use them on 3D objects. To copy translate or rotate values from points:

1. Select one or more points in your footage.
2. Right-click on them and select **copy > translate** or **ZXY rotate**.
3. Go to the control panel of your 3D object, right-click on the Animation menu and select **Paste**.

Exporting a Point Cloud

You can export a point cloud as an FBX file using WriteGeo node. Note that you should only render your exported point cloud for one frame, since a point cloud generated with either the CameraTracker node or the PointCloud Generator node doesn't change over time by default.

1. Create a WriteGeo node and attach it to your CameraTrackerPointCloud node.
2. In the WriteGeo controls, select **fbx** in the **file type** drop-down, and make sure you have the **point clouds** box checked.
3. Browse to the destination you want to save the .fbx file in the **file** field and give your object a name.
4. Click **Execute**.
5. In the pop-up dialog, specify the frame range you want to include in the file and click **OK**.

   A progress bar will display, and an .obj file is saved.

Tracking Multiview Projects

You can use the CameraTracker node in a stereoscopic or multiview project. The workflow is largely the same as in a single view project, with only a few differences:

1. Connect the CameraTracker node as described in "Connecting the CameraTracker Node" on page 662.
2. Use the **Principal view** dropdown on the **CameraTracker** tab to select the view to select your primary view.
3. Follow the workflow described for setting tracking features in "Tracking Features in a Sequence" on page 662.
4. See "Setting the Camera Parameters" on page 667 and in addition, on the **Solver** tab, under **Solver constraints**, define the constraints on your stereo cameras:
   - Check **Aligned Stereo Cameras** to automatically align the position of your cameras.

- Check **Constant Interaxial Distance** to confirm that the two stereo cameras are at the same distance from each other throughout the footage
- Check **Constant Interaxial Convergence** to confirm that the angle at which the two stereo cameras are set is constant throughout the footage.

5. Then continue to further edit your results. For more information, see "Solving the Camera Position" on page 669, "Deleting Tracks" on page 670, and "Using the Point Cloud" on page 675.
6. To visualize both cameras for your views, you can also choose to create a multi-view rig with a Camera for each of your views by clicking **Create Rig** on the **CameraTracker** tab.

Tip *Make sure you check that the tracks in your multiview project are matched correctly between each view and either manually delete bad tracks or mask out regions that track poorly. For more on deleting tracks, see "To delete tracked features" on page 671 and on masking out regions of your image, see "Masking Out Regions of the Image" on page 664.*

2 ADDING AND REMOVING LENS DISTORTION

Nuke's LensDistortion node allows you to distort or undistort an image according to a radial distortion model.

Quick Start

To get you started with using LensDistortion, here's the workflow in a nutshell.

1. Read in an input sequence, connect it to a Lens Distortion node (**Transform** > **LensDistortion**), and connect the output to a Viewer.
2. To apply lens distortion to the input manually, change the parameters on the **LensDistortion** tab. Turn on **Undistort** to invert the current distortion. For more information, see "Adjusting LensDistortion Parameters" on page 683.
3. To estimate the lens distortion on the input, you have three options. Each one of these will calculate the distortion present and set the values in the **LensDistortion** tab.
   - **Image Analysis:** use this option to estimate the distortion automatically without the help of grids or lines. Image analysis tracks features through the sequence and finds the distortion model that best describes the way the same 3D structure is projected onto different parts of the image. For more information, see "Calculating Lens Distortion Automatically" on page 681.
   - **Grid Analysis:** use this option to estimate the distortion from a checkerboard or thin line grid, for greater accuracy. For more information, see "Analyzing Distortion Using a Grid" on page 682.
   - **Line Analysis:** use this option to estimate the distortion from lines drawn along features in the input that are known to be straight. For more information, see "Analyzing Distortion Using Lines" on page 682.
4. If you want, you can also calculate the lens distortion on one image and apply that distortion to another image with the help of an STMap node. For more information, see "Calculating the Distortion on One Image and Applying it to Another" on page 685.
5. You can use the estimated lens distortion values on the LensDistortion tab to distort an image on a Card. For more information, see "Applying Lens Distortion to a Card Node" on page 685.

Note *When using a ScanlineRender node downstream from a LensDistortion node, by default, the ScanlineRender node does not pull pixels from outside the frame and you may end up with parts of the image missing. To fix this, you* *can use the* ***overscan*** *slider in the ScanlineRender controls to set the maximum additional pixels to render beyond the left/right and top/bottom of the frame. For more information on the ScanlineRender node and the overscan control, see "To render pixels beyond the edges of the frame:" on page 493.*

Calculating Lens Distortion Automatically

Image analysis estimates the lens distortion in a sequence (and sequence only, you can't perform this analysis on a still image) automatically. It tracks features through the sequence and finds the distortion model that best describes the way the same 3D structure is projected onto different parts of the image. To analyze lens distortion automatically, do the following:

1. Click the **Image Analysis** tab.
2. If your sequence is long, you might wish to select the range of frames to analyze by choosing **Specified Range** under **Analysis Range**. Another reason to change the range would be to select a section of the sequence that has some strong features for it to track, or where the features it detects are well-distributed across the image (rather than being all clustered together in one corner, for example). Specify the range in the **Analysis Start** and **Analysis Stop** fields.
3. If you don't want to analyze all regions of your image, for example if there are moving objects in the image, you can provide a matte in the **Mask** input or the alpha channel of the **Source** input. From the **Mask Channel** dropdown menu, select the matte channel.
4. Finally, press **Analyze Image** to start the estimation.
5. When the analysis is finished, the distortion parameters will be set on the **LensDistortion** tab to undistort the input.

Image Analysis Parameters

- **Analysis Range** – Choose whether to analyze the whole of the **Source Clip Range** or a **Specified Range**, defined in the **Analysis Start** and **Analysis Stop** fields.
- **Camera Motion** – Select **Rotation Only** if this applies, or **Free Motion** for all other types of motion.
- **Mask Channel** - If you don't want to analyze all regions of your image, for example if there are moving objects in the image, you can provide a matte in the **Mask** or **Source** input and use this control to select the matte channel.
- **Analyze Image** – Track features through the input sequence and estimate the lens distortion present. When the analysis is complete, this will set the parameters on the **LensDistortion** tab in order to undistort the sequence.

- **Feature Overlay** – After analyzing, you can choose to display the **Features** or **Features and Tracks** that were detected on the current frame.

Analyzing Distortion Using a Grid

Grid analysis estimates the distortion from a checkerboard or thin line grid, for greater accuracy. As a general rule, if you have a grid you can use to calculate your lens distortion, you should use grid analysis to do this. To analyze a grid, do the following:

1. To estimate the distortion from a grid, turn to the **Grid Analysis** tab.
2. Connect a grid to the LensDistortion node's **Source** input and select the appropriate **Grid Type**.
3. Press **Analyze Grid** to estimate the distortion and undistort the grid.
4. After analyzing a grid, you can check **Align Grid** to correct for perspective distortion caused by the grid not being shot square-on to the camera.
5. If you want to apply the grid's distortion to an image input, disconnect the grid from the **Source** input and connect your image to it instead.

Grid Analysis Parameters

- **Grid Type** – Choose the type of grid you want to use, **Checkerboard** or **Thin Line** grid.
- **Analyze Grid** – Calculate the lens distortion present. When the analysis is complete, this will set the parameters on the **LensDistortion** tab in order to undistort the sequence.
- **Align Grid** - Check to correct for perspective distortion caused by the grid not being shot square-on to the camera. In other words, you can align the grid lines with the image edges.
- **Grid Overlay** – After analysis, you can choose to display the **Original** or **Undistorted Grid** lines over the image.

Analyzing Distortion Using Lines

Line analysis estimates the distortion from lines drawn manually along features in the input that are known to be straight. This can be useful if you have a single image to undistort and no grid available, and can't therefore use grid analysis or image analysis. Another case for using line analysis might be if you have a grid for your sequence but the grid analysis failed, for instance due to bad lighting. To analyze lens distortion using lines, do the following:

1. Click the **Line Analysis** tab and check the **Drawing Mode On** box.

2. Draw a line along a feature in the image that you know should be straight (if there were no distortion present) by left-clicking along the feature at intervals. At each click, points appear on the image. You can move a point by dragging it.
3. Join the points into a line by right-clicking on the image. This will not create a point.
4. Repeat steps 2 and 3 until you have at least three lines.
5. Press **Analyze Lines** to estimate the lens distortion and undistort the image.

Line Analysis Parameters

- **Drawing Mode On** – Check this box to begin drawing lines along features in the input that should be straight. Draw a line by left-clicking along the feature at intervals and finish a line by clicking the right mouse button. You need at least three left-clicks to make one line, and at least three lines to initiate the distortion.
- **Distortion Terms** – Choose the number of distortion terms to estimate. With only a few lines, sometimes better results can be achieved by estimating only the first radial distortion term.
- **Estimate Center** – Choose whether you want to estimate the distortion center. If the lines are not evenly distributed around the image, the center estimate is unlikely to be very good and you may get better results to approximate it by using the image center instead (the default).
- **Analyze Lines** – Click to estimate the lens distortion from the current lines. When the analysis is complete, this will set the parameters on the **LensDistortion** tab in order to undistort the sequence.
- **Delete Last Line** – Delete the last line drawn. Repeat to delete the previous line, and so on.
- **Delete Last Point** – Delete the last point drawn. Repeat to delete previous points that are not yet part of a completed line.
- **Hide Lines** – Hide the line overlay.
- **Clear All** – Delete all lines and points.

Adjusting LensDistortion Parameters

After you've estimated your lens distortion, you can view the values on the **LensDistortion** tab. You can also change these to apply distortion to the input directly.

- **Output Type** - Choose your output type depending on whether you want to distort the input image directly or just preserve the distortion information to use it on another image.
  - **Image** - Choose to distort or undistort the input image.

  - **Displacement** - Choose to leave the input image alone and store the pixel displacements corresponding to the calculated distortion in the UV channels so they can be used by the STMap node. Choosing the **Displacement** option enables you to analyze the distortion on one image or grid and use the STMap node to apply that distortion to another image. For more information, see "Calculating the Distortion on One Image and Applying it to Another" on page 685.
- **Lens Type** – Select your lens type. You can choose between **Spherical** and **Anamorphic** lenses.
- **Radial Distortion 1** – Define the first radial distortion term. This is proportional to r^2, where r is the distance from the distortion center. You can define this manually, or use image analysis, grid analysis, or line analysis to fill it out automatically.
- **Radial Distortion 2** – Define the second radial distortion term, proportional to r^4. You can define this manually, or use image analysis, grid analysis, or line analysis to fill it out automatically.
- **Distortion Center** – Define the values for the center of the radial distortion.
- **Anamorphic Squeeze** – Define the anamorphic squeeze value. If you select an anamorphic lens, the distortion in x will be scaled by this amount.
- **Asymmetric Distortion** - This is enabled only for anamorphic lenses. Enter values to define asymmetric distortion to correct for slight misalignments between multiple elements in the lens.
- **Analyze from Current Lens** – Check to analyze the distortion from the current values. When this is selected, the Image Analysis and Grid Analysis modes will skip their initial estimation steps and instead use the current parameter values as an initial guess from which to estimate the distortion. If you already have a rough idea of the distortion present, this can speed up the estimation, particularly for the Image Analysis mode.
- **Undistort** – Check to apply the inverse of the current distortion.
- **Reset** – Click to reset the parameters for zero distortion.
- **filter** - Choose a filtering option. For more information on the filtering options, see "Choosing a Filtering Algorithm" on page 237.
- **Distortion Scaling** - Choose either **Scale to Input Format** or **Choose Format** depending on whether you want to scale to your input format or choose another format. By default the distortion is scaled according to the input format size. You might however want to select a different format, for example, if you have previously undistorted an image and changed the format to match the undistorted image size, then want to reapply the distortion later.

- **Scale to Format** - If you selected **Choose Format** next to the Distortion Scaling control, you can use this dropdown menu to select a format that you want to use to scale your image.
- **Card Parameters** – Fill in **a**, **b** and **c** values in the Card Parameters folder. These correspond to the parameters on the Lens Distortion tab of Nuke's Card node. For more information, see "Applying Lens Distortion to a Card Node" on page 685.

Calculating the Distortion on One Image and Applying it to Another

You can use the LensDistortion node to analyze the distortion on one image or grid, and then apply that distortion to another image. Do the following:

1. Use the LensDistortion node to analyze an image or grid to get the distortion parameters. Make sure **Output Type** is set to **Image** on the **LensDistortion** tab of the LensDistortion node. Check that the resulting image looks undistorted.
2. Set **Output Type** to **Displacement**.

   In this mode, the LensDistortion node leaves the input image alone and writes the pixel displacements that correspond to the calculated distortion to the UV channels. These appear as forward **motion** channels in the Viewer's channel dropdown. The displacement is in the format used by the STMap node.
3. Uncheck **Undistort** to apply distortion instead of removing it.
4. Select **Transform** > **STMap** in the Nuke toolbar to insert an STMap node in the Node Graph. Connect the STMap node's **stmap** input to the LensDistortion node.
5. Import an image you want to distort with the distortion calculated in step 1. Connect that image to the **src** input of the STMap node.
6. In the STMap controls, select **motion** from the UV channels dropdown menu. This should now apply the distortion from the LensDistortion node's input to the STMap node's **src** input.

Applying Lens Distortion to a Card Node

You can use the calculated lens distortion values to distort an image projected onto a card in 3D. Do the following:

1. Open the **Card Parameters** folder on the LensDistortion tab.
2. Hold down **Ctrl/Cmd** and drag the **x**, **y** and **z** values from the **scale** parameter's values on to the corresponding fields on the **Card** tab in the Card node (**3D** > **Geometry** > **Card**) control panel. This will set the right scale for the reproduced distortion on the Card.
3. Hold down **Ctrl/Cmd** and drag the **a**, **b,** and **c** values onto the corresponding fields on the **Lens Distortion** tab in the Card node control

panel. These values are now used to undistort the image attached to the **img** input of the Card node.

Note *The Card node's lens distortion parameters are estimated and won't necessarily match the LensDistortion results exactly, particularly if the calculated lens is asymmetric, in other words having different amounts of distortion on the x and y axes.*

3 CREATING DENSE POINT CLOUDS AND 3D MESHES

Dense point clouds are a useful starting point for 3D modeling and can be helpful in positioning 3D objects into a scene. Using the PointCloudGenerator node you can create a dense point cloud based on the information generated by CameraTracker and use the dense point cloud further to create a 3D mesh of your 2D footage with the PoissonMesh node.

Quick Start

To get started quickly on dense point clouds and meshes, here's the workflow in a nutshell:

1. Make sure your PointCloudGenerator node is connected to the appropriate footage and Camera node. For more information, see "Connecting the PointCloudGenerator Node" on page 687.
2. Track the footage and solved camera to create a dense point cloud. For more information, see "Tracking a Dense Point Cloud" on page 688.
3. Filter the result to adjust it and eliminate bad points. For more information, see "Filtering Your Point Cloud" on page 689.
4. Then if you need to, you can move on to creating a mesh with the PoissonMesh node. For more information, see "Creating a Mesh Using a Point Cloud" on page 689.

Connecting the PointCloudGenerator Node

The PointCloudGenerator node is often created automatically by the CameraTracker node in the process of creating a scene, after tracking and solving your 2D footage. In order to work, the PointCloudGenerator needs a Camera node with the solved camera path and a 2D footage or a CameraTracker node. To connect the node, do the following:

1. Connect the necessary nodes to the inputs:
   - **Source** - Connect your footage or the CameraTracker node you used to track it here.
   - **Camera** - Connect the solved Camera node here.
   - **Mask** - Connect your optional mask for excluding areas in the footage you don't want tracked.
2. If necessary, set **Mask** to the component you want to use as a mask:
   - **None** - Track features in the whole footage.
   - **Source Alpha** - Use the alpha channel of the source clip to define which areas to ignore.

  - **Source Inverted Alpha** - Use the inverted alpha channel of the source clip to define which areas to ignore.
  - **Mask Luminance** - Use the luminance of the mask input to define which areas to ignore.
  - **Mask Inverted Luminance** - Use the inverted luminance of the mask input to define which areas to ignore.
  - **Mask Alpha** - Use the mask input alpha channel to define which areas to ignore.
  - **Mask Inverted Alpha** - Use the inverted mask input alpha channel to define which areas to ignore.

3. You can also change the value of **Point Size** under **Display** to adjust the pixel size of the points in your cloud.

Tracking a Dense Point Cloud

The first step towards creating a dense point cloud is to track your footage for more 3D feature points using the information from the footage, CameraTracker and the solved Camera. Do the following:

1. Set the controls for selecting keyframes:
   - **Automatic Keyframe Selection -** Check to automatically select keyframes for tracking and point cloud creation. Keyframes are placed where there is camera parallax and points are tracked across keyframes to triangulate a 3D position.
   - **Angular Threshold** - Select where the triangulation angle exceeds a certain threshold. Use a larger angle to generate keyframes with a larger camera parallax.
   - **Keyframes** - Adjust where the keyframes are set. If you unchecked the **Automatic Keyframe Selection**, you can use this control to manually set the keyframes using the **Add** and **Delete** buttons.
   - **Frame Spacing** - You can also set your keyframes manually by setting the spacing for the keyframes here, and clicking **Add All** or **Delete All**. For example, if you set this value to 2 and click **Add All**, keyframes will be set to every other frame throughout your sequence.
2. Set controls for **Dense Tracking:**
   - **Point Separation** - Set the separation value in pixels for points in the dense point cloud.
   - **Track Threshold** - Set a threshold value between 0 and 1. This threshold controls how similar features look over a number of frames. You can adjust this value to test whether a track is reliable.
3. Click **Track Features** under **Analysis**. The PointCloudGenerator node tracks the sequence and creates a dense point cloud.

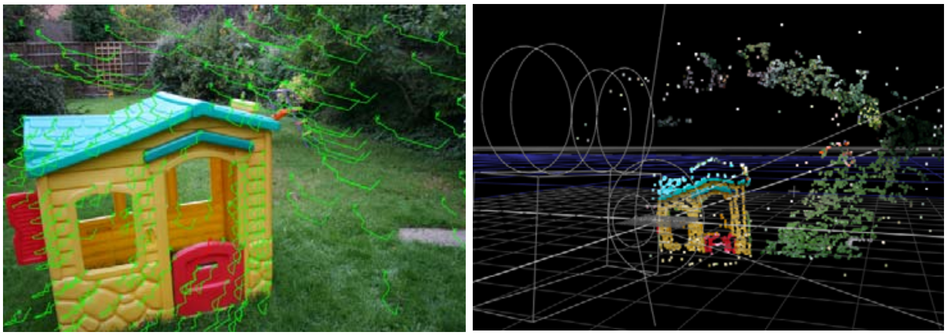


Figure 3.1: Dense point cloud points in 2D and 3D view

Filtering Your Point Cloud

You can filter your point cloud and thus adjust the number and quality of points it includes. NukeX filters the tracked point cloud directly making your work faster as there's no need to retrack to adjust your cloud. To filter your cloud, adust the following controls and view the result in the 3D Viewer:

- **Filter Iterations** - Set the number of filtering iterations. Each filtering iteration removes some points, which in turn removes support for other incorrect points. Filtering more times gradually erodes noisy points by getting rid of a few each iteration.
- **Tension** - Set the tensions between points. Increase to smooth out points and remove noise.
- **Smoothness** - Set a tolerance for the points on a plane to be offset from the plane. This controls how much error is allowed from a point to a locally fitted plane. Points that differ from their local plane by more than this tolerance are rejected.
- **Smoothing Points** - Number of points used to fit a local plane which measures local smoothness within the **Smoothness** tolerance value.
- **Angle Threshold** - Set the threshold for the minimum angle to triangulate 3D points (in degrees). Points with a large triangulation angle are more accurate. Set a threshold of 0 to triangulate all points. Increase the threshold to retain the more accurate points. As a rule of thumb, anything below 5 degrees is likely to be incorrect.

Creating a Mesh Using a Point Cloud

The PoissonMesh node uses information from a dense point cloud to generate a mesh that you can further use as a 3D object, in 3D modeling for instance. The PoissonMesh node is based on the Poisson Surface Reconstruction calculation method. The original source code and paper were created by Michael Kazhdan, Matthew Bolitho and Hugues Hoppe (for more information, see http://www.cs.jhu.edu/~misha/Code/PoissonRecon/).

Before using the PoissonMesh node, you need a dense point cloud with normals such as the one created by the PointCloudGenerator node. For more information on creating a dense point cloud, see "Tracking a Dense Point Cloud" on page 688. When you've got one, you're ready to create a mesh:

1. Connect the PoissonMesh node to a PointCloudGenerator node.
2. A mesh is created automatically using the information from the PointCloudGenerator node and you can view it in the 3D view.
3. If you want to, you can tell PoissonMesh to use the selected points only in the point cloud to create a mesh. To do this, check the **Use Selection** box in the control panel. You can also use the **Filtering** box to use the PointCloudGenerator filtering to refine your mesh.
4. You can also adjust other properties for your mesh on the **PoissonMesh** tab:
    - **Depth** - set the maximum depth (in integers) for the calculation tree used to create your mesh. You can enter values up to 10, but increasing this value increases the solver's memory usage, so it's a good idea to keep this as low as possible.
    - **Scale** - set the ratio between the 3D cube used to create the mesh and the bounding box of the footage it's created from.
    - **Solver Divide** - set the depth at which the solver equation is used in the creation of your mesh. This can help you reduce your memory usage when generating complex meshes.
    - **Iso Divide** - set the depth at which the iso-surface extractor is used in extraction. This can help you reduce your memory usage when generating complex meshes.
    - **Samples per Node** - set the minimum number of sample points used to create your mesh. Use larger values if your original footage is very noisy.
    - **Confidence** - check to use the size of the normals as confidence information when generating the mesh. This may take longer, but gives you better results since your point cloud point creation gets double-checked.

Note *You can only use the PoissonMesh node in NukeX as the information used to create the mesh is not stored in the node and therefore cannot be recreated for viewing in Nuke.*

Adding Texture To The Point Cloud Mesh

You can add texture to your point cloud mesh using the Project3D and ApplyMaterial nodes. Do the following:

1. Connect your texture node to the Project3D node.

2. Then connect the Project3D node to the **mat** input of the ApplyMaterial.
3. Connect the other input of the ApplyMaterial node into the PoissonMesh node. The mesh is now textured with your texture.

Tip *You can export both your point cloud and the mesh as FBX objects to other applications, but don't forget to export the Camera node too. For more information about exporting FBX files, see "Exporting Geometry, Cameras, Lights, Axes, or Point Clouds" on page 491.*

4 GENERATING DEPTH MAPS

You can use the DepthGenerator node to generate a depth map (or a z-depth map) from your footage. The node uses information from a CameraTracker node to create a channel that displays variations in depth.

A depth map is an image that uses the brightness of each pixel to specify the distance between the 3D scene point and the virtual camera used to capture the scene. You may need a depth map, for example, if you want to introduce fog and depth-of-field effects into a shot. In Nuke, the ZBlur node (**Filter > ZBlur**) requires a depth map in its input.

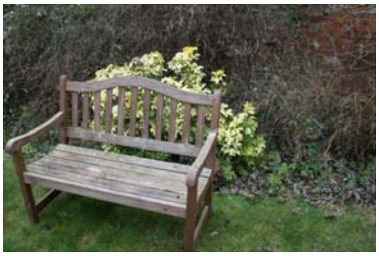


Figure 4.1: Source image.

Figure 4.2: Depth map.

How is the Depth Calculated?

The DepthGenerator calculates the depth per-pixel in the input image as Z = 1/d where d is the depth from the center of the camera along the camera viewing axis. Each pixel in an image corresponds to a ray that is formed by projecting a 3D point (x,y,d) down to the 2D image plane (u,v) as shown in Figure 4.3. A 3D point is recovered from a pixel by first forming a ray for the pixel which is defined by the camera focal length f and center-point (cx,cy), then setting the depth d of the point from the camera.

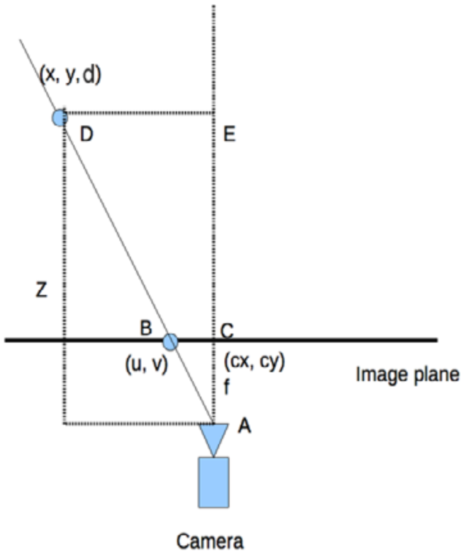


Figure 4.3: Calculating the depth.

Connecting DepthGenerator

To use the DepthGenerator node, you first need to analyze your footage with the CameraTracker node. Using the Camera node created by CameraTracker, DepthGenerator can then create a depth map from your footage.

To Connect the DepthGenerator Node

1. Click **3D** > **CameraTracker** to create a CameraTracker node and connect it to your footage. DepthGenerator can only create a depth map from a sequence, so you can't use a still image.
2. Track features in your footage, solve the camera, and create a scene using the CameraTracker node. For more information on using the CameraTracker node, see "Camera Tracking" on page 661.
3. Create a DepthGenerator node by clicking **3D** > **DepthGenerator.**
4. Connect the **Source** input to your footage and the **Camera** input to the Camera node created by the CameraTracker node.

Generating a Depth Map

1. Connect the DepthGenerator node as described above.

   You can now view the depth channel by selecting it from the list of channels in the Viewer.

2. Use the **Frame Separation** control in the control panel to select the offset between the current frame and the frame against which to calculate depth for your footage. For example, if your current frame is 100, and your frame separation is 2, DepthGenerator will use frame 100 and frame 102 to generate the depth map.

   Ideally, you want frames close together so that the images are similar and include the same bits of the world. However, you get more accurate depth when the frames are further apart and the "baseline" between the cameras is bigger. So, for fast camera moves, you will need a small frame separation, and for slow camera moves, you can use a larger frame separation. To change the separation for fast and slow movements, you can animate the **Frame Separation** control.

   Note that this control does NOT affect the number of frames used in the calculation, as that is always 2 (or 3, if **Bidirectional** is selected).
3. Use the **Bidirectional** control in the control panel to choose whether you want to calculate depth using Frame Separation values to both directions, backwards and forwards, in the sequence. For example, if **Bidirectional** is selected, and your current frame is 100, DepthGenerator uses frame 98, frame 100 and frame 102 to generate the depth map. Using Bidirectional calculation is usually slower, but produces a more accurate result.
4. In addition to camera information from the CameraTracker node, DepthGenerator needs a disparity field to calculate the depth map. It creates this disparity field automatically for the frames you've selected to use. The disparity field maps the location of a pixel in the current frame to the location of its corresponding pixel in the other frame(s). If necessary, you can adjust the following Disparity Generation controls:

   - **Disparity Field Detail** - Adjust this to vary the resolution of the images used to calculate the disparity field. The larger the value, the greater the processing time, but the more detailed the vectors. The default value of 0.5 will generate a vector at every other pixel.
   - **Smoothness** - Set the smoothness weighting of the disparity calculation. Disparity fields usually have two important qualities: they should accurately match similar pixels in one frame to another and they should be smooth rather than noisy. Often, it is necessary to trade one of these qualities off against the other. A high smoothness will miss lots of local detail, but is less likely to provide you with the odd spurious vector. A low smoothness will concentrate on detail matching, even if the resulting field is jagged. The default value of 0.5 should work well for most sequences. However, if your scene is planar or near planar, you can try increasing the value.
   - **Occlusions** - Choose the algorithm used to create the disparity field. In most cases, **Normal** is faster and produces more accurate results than

Severe. However, if there are large occluded areas between the frames that are used to generate the depth map, **Severe** may produce better results than **Normal**.

- **Mark Bad Regions** - Check to mark the regions where the depth calculation is ambiguous. These regions might occur if CameraTracker hasn't been able to calculate the depth of all pixels using the camera data, due to certain camera movements for instance.

5. To view your results more easily, hold down **Shift** and select **Color** > **Math** > **Multiply** to add a Multiply node in a new branch after DepthGenerator. In the Multiply controls, set **channels** to **depth** and adjust **value**.

   Similarly, add a Gamma node (**Color** > **Math** > **Gamma**) after the Multiply node. In the Gamma controls, set **channels** to **depth** and adjust **value**.

Figure 4.4: Depth map.

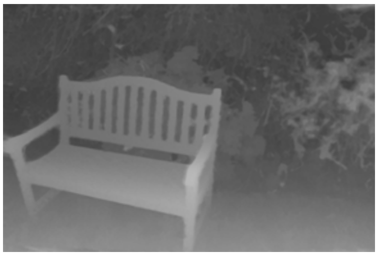


Figure 4.5: Depth map after the Multiply and Gamma nodes.

5 USING THE MODELER NODE

The Modeler node is used to create flat polygonal geometry using a tracked camera and an input image clip for visual reference. You can also use other 3D geometry and point clouds with the Modeler, if you have these created for your scene.

Quick Start

To get you started using the Modeler quickly, here's the gist of it:

1. Connect the Modeler node to your footage and a changing Camera. For more information, see "Connecting The Modeler Node" on page 696.
2. Use the **Add Faces** tool to draw a face on your 2D footage, then scrub to another frame and move the face to its place, using the **Select/Edit** tool. For more information, see "Adding Faces" on page 696.
3. Right-click on an edge to split it and use the **Add Vertex** tool to add single vertices on your footage. For more information, see"Editing Faces and Vertices" on page 698 and "Adding Single Vertices" on page 699.
4. Move and extrude faces with the **Select/Edit** tool. For more information, see"Moving and Extruding Faces" on page 698.

Connecting The Modeler Node

1. Create a Camera to match with your footage using the CameraTracker node. The Modeler needs a changing Camera (with change of more than 5 degrees) in order to calculate 3D points from your 2D footage. For more information, see "Solving the Camera Position" on page 669.
2. Connect the Modeler node (**3D** > **Geometry** > **Modeler**) to the Camera and the source footage.
3. You can further adjust the rotation, translation and other similar values of your 3D world using the controls on the **Modeler** tab. For more information on using these controls, have a look at "Transforming from the Node Properties Panel" on page 459.

Adding Faces

1. Select the **Add Faces** tool (or press **A**) in the Modeler toolbar on the left side of the Viewer.
2. Add at least three points in the Viewer to make up a polygon. Finish the polygon by clicking on the first point, or press **Enter** to close it automatically. The Modeler automatically creates a 2D face using these points and activates the **Select/Edit** tool.

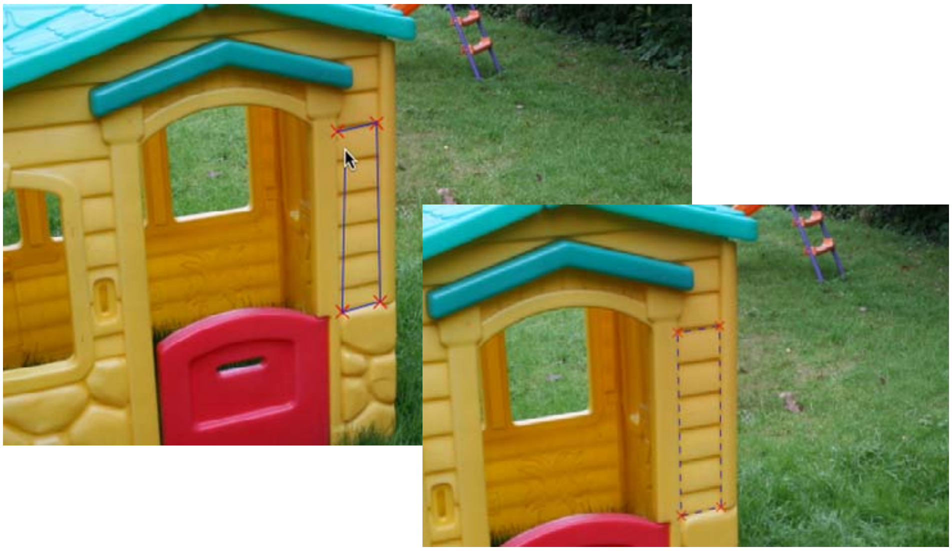


Figure 5.1: Drawing a face with the Modeler node

3. Scrub to another frame in the timeline. The Modeler needs to know the positions of your points in at least two different keyframes in order to calculate the 3D points.
4. Move the vertices that make up the polygon to match their positions in the current frame. The Modeler automatically calculates a 3D point to match with each of the 2D points.
5. You can now view your face in the 3D Viewer mode. The Modeler also adds blue crosshairs to mark the calculated positions of your vertices. You can view your original, manually set vertices as red crosshairs that are connected to their blue counterpoints with a red line. Note that you can also change the colors of the lines and crosshairs on the **Options** tab.

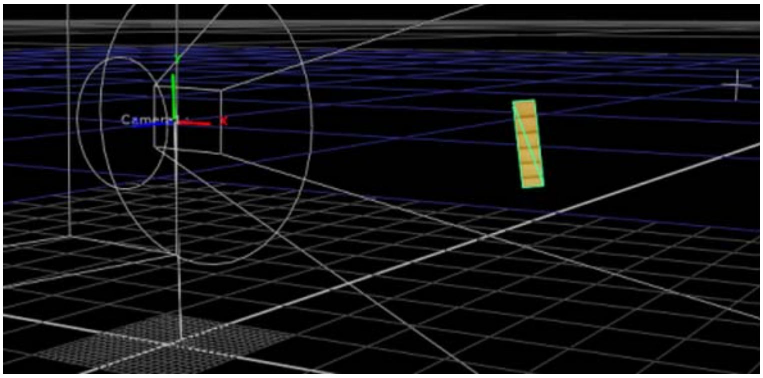


Figure 5.2: A face in 3D view

Editing Faces and Vertices

1. When you're finished with drawing a face, you can use the **Select/Edit** tool (hotkey **E**) to adjust it in the Viewer. You can also use the number keys on the numeric keypad to nudge vertices.
2. You can right-click on a vertex and select **delete vertex** or **delete face** to remove the vertex or the entire face. If you remove a face, the vertices remain as single vertices that don't make up a face.
3. If you select one edge of your polygon, you can right-click on that and select **split edge** to split it in half. The Modeler adds a new vertex on the selected edge.
4. You can also view your faces and their plane values on the **Faces** tab of the Modeler control panel. Using the **Faces** table, you can select your faces and rename them by double-clicking in the **plane name** column. You can also delete your faces by selecting them and clicking the **Delete Face** button, and make them visible or invisible by toggling the **visible** box.
5. On the **Vertices** tab you can view a table of your individual vertices and edit their x, y and z values. Normally, however, you don't need to adjust this table at all.

Tip *You can link values into the vertex table, that is set expression values in the track x and track y columns. To do this, right-click on the vertex table cell you want, and select **Edit Expressions**, or just press the equals sign (=) key on the keyboard. The Edit Expressions dialog opens and you can edit expressions for all the values for that row in the table.*

To link values out of the table, in other words access table values from another expression in another control, use the following expression:

Modeler1.vertexList.1.track_y

*This will give you the value of the **track y** column on the first row of the table.*

Moving and Extruding Faces

You can move or extrude a face by selecting the **Select/Edit** tool.

1. Select the **Select/Edit** tool (hotkey **E**) from the Modeler toolbar.
2. Hold **Cmd/Ctrl** and click to select the required face, or select the face from the list on the Modeler node's **Faces** tab.
3. To move a face, grab the handle in the middle of the face in the 2D view and push or pull the face.
4. To extrude a face, **Ctrl/Cmd** + drag the handle in the middle of the face on the 2D view to stretch it into a three dimensional polygon. The Modeler creates faces to complete the object, for example a cube from a square face.

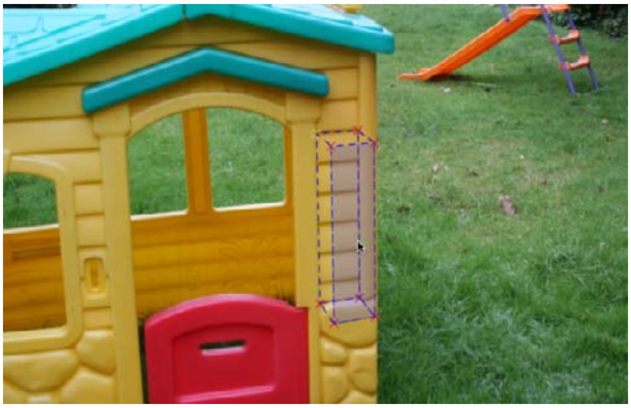


Figure 5.3: Extruding a face

5. If you extrude the same face again by **Ctrl/Cmd**+dragging it, the Modeler creates another three dimensional polygon next to your first one.

Adding Single Vertices

You can add single vertices on your footage too. The Modeler calculates 3D points for them, but doesn't make them into faces. Single vertices can be useful with, for example, positioning 3D objects.

1. Select the **Add vertices** tool or use the hotkey **V**. You can also add vertices in the Viewer by right-clicking and selecting **add vertex**.
2. Click in the Viewer to add single vertices on your footage.
3. Scrub to another frame in the timeline, and move the vertices to match their positions in the current frame. The Modeler calculates 3D points for your vertices on the **Vertices** tab.

Adjusting Modeler Options

On the **Options** tab, you can change the colors Modeler uses to mark your vertices, faces and so on. Under **Colors** you can adjust:

- **selection color** - to change the color of a selected vertex or edge.
- **track vertex color** - to change the color of the vertices you've set manually.
- **line color** - to change the color of an unselected edge or a calculated vertex.
- **inverse line color** - check to use a line color that is the inverse of the part of the image that the line is over.
- **selected fill color** - to change the color of a selected face.
- **selected line color** - to change the color of edges in a selected face.

Tip *If you want to change the sizes of the vertex crosshairs or vetex pick sizes for instance, you can change them in **Preferences** > **Viewers** > **Interaction**.*

6 USING PROJECTIONSOLVER

ProjectionSolver helps you align 2D with 3D points. You can for instance align 2D images with a 3D model. The output from the ProjectionSolver node is a Camera node that matches with your 2D footage's camera movement.

Quick Start

To get you swiftly on your way with using ProjectionSolver, here's the workflow in a nutshell:

1. Connect your ProjectionSolver, Read, ReadGeo and other optional nodes. For more information, see "Connecting the ProjectionSolver Node" on page 701.
2. Match your 2D footage and 3D model with locators (at least 6) and create keyframes for their movement through the frames of your footage. For more information, see "Matching 3D Geometry with 2D Footage" on page 702.
3. Click **Solve Camera**. See "Solving Your Camera" on page 703.
4. Refine your scene in the Viewer and use the **Refine Output** and **Live Update** controls to view the changes. For more information, see "Refining the Solve" on page 704.
5. When you're happy with your solve, click **Create Camera**.
6. You can also create Card nodes by clicking **Create Cards** to help you visualise your scene without scrubbing in the timeline. For more information, see "Creating Cards" on page 705.

Connecting the ProjectionSolver Node

To start using ProjectionSolver, you first need a 3D object and the 2D images you want to project onto the object. Then:

1. Create a ProjectionSolver node by clicking **3D** > **ProjectionSolver**.
2. Connect your 3D model to the **Geo** input.
3. Connect your 2D footage to the **Source** input of the ProjectionSolver node.

The output from the ProjectionSolver node is a Camera node that matches with your 2D footage's camera movement. You can also create Cards to match with each different camera position.

Matching 3D Geometry with 2D Footage

First, you need to tell ProjectionSolver which points in the 2D footage go with their counterparts on your 3D model. Do the following:

1. Use the 3D vertex selection tool in the Viewer to select a vertex on the 3D model.
2. Go to your 2D footage and right-click **add locator** on the 2D point for the vertex.

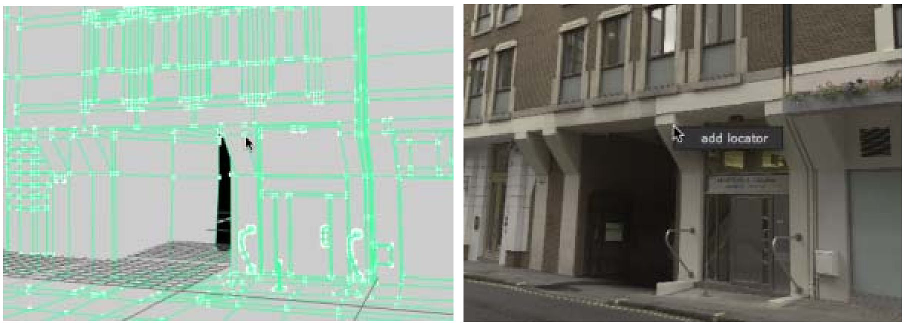


Figure 6.1: Selecting a vertex in 3D (left) and adding a locator in 2D.

3. Repeat steps 1 and 2 until you have at least 6 locators on the frame.
4. Create keyframes for the locators under the **Locators** tab. Click the Animation menu for each 2D locator and select **set key**. You can also click the **Key All Locators** button to key all your locators at once. Note that you don't have to create keys to solve a single frame.
5. Go to the next frame and move the 2D locators to the correct position in the Viewer, thereby creating a 2D track.

You should add at least 6 locator points to match the 2D footage and 3D model with each other. You get a much better result if you create 8 locator points though, and if you want to ensure great results, you'll want to create about 10-12 locator points.

Tip *When solving multiple frames, the locators do not have to be keyed at every frame. Only the keyed locators will be used to solve each frame.*

Copying Values From a Tracker Node

You can also use the Tracker node to get tracking information for a locator point.

1. Track a point with the Tracker node.
2. Click on the tracked point's Animation menu and select **Copy** > **Copy Values**. Note that you should only copy the tracking values, not use the **Link to** option.

3. Go to ProjectionSolver control panel and click **Add Locator** on the **Locators** tab.
4. Click the Animation menu of the new locator and select **Copy** > **Paste**.

Solving Your Camera

1. After you've created your locators, you can select which locators are to be used in the solve by toggling the **Active** box next to each locator on the **Locators** tab.
2. On the **ProjectionSolver** tab, click **Solve Camera**. ProjectionSolver solves your camera and gives you a solve error value in the **Solve Error** field. Should any of the frames in your footage fail to solve, you'll see a message box with tips that might help. You can also try adjusting the **Solver** and **Lens** controls for better results.

Adjusting Solver and Lens Controls

Under **Solver**, you can adjust the parameters that the ProjectionSolver uses for solving the camera. Adjust:

- **Focal Length Type** - Set the focal length for the camera.
  - **Unknown Constant** - Select this if there's no zoom and the focal length is unknown.
  - **Unknown Varying** - Select this if the focal length is unknown and changes.
  - **Approximate Constant** - Select this if there is no zoom and an approximate focal length is available and enter the focal length value next to **Focal Length**.
  - **Approximate Varying** - Select this if an approximate focal length is available and changes and enter the focal length value next to **Focal Length**.
  - **Known** - Select this if the focal length is defined for the sequence and enter the focal length value next to **Focal Length**.
- **Focal Length** - Set the focal length for approximate and known solves. You can animate focal length to define a varying focal length. The units should match the units used for the film back size (millimeters, for example).
- **Film Back Size** - Set the size of the imaging sensor of the camera and specify the units you want to use by selecting either **millimeters** or **inches** in the **Units** drop-down. You only need to specify this if you have also specified focal length. The units should match the units used for the focal length.

Under **Lens**, you can adjust the parameters ProjectionSolver uses to account for lens distortion in your camera:

- **Lens Distortion** - select the type of distortion to expect.
  - **No Lens Distortion** disables all lens distortion controls and treats the footage as having no distortion. This option gives the best solve to work on the original footage. The calculated focal length will compensate for lens distortion.
  - **Known Lens** allows you to specify the lens distortion manually and to use that information in doing the camera solve. Use this when you have a grid available for defining lens distortion. You can calculate the distortion parameters in a LensDistortion node first and paste the values onto ProjectionSolver.
  - **Refine Known Lens** gives you a chance to give an approximate distortion which to use that is refined in the camera solve.
  - **Unknown Lens** calculates the lens distortion automatically from the sequence (in the same way as the **Image Analysis** option in the LensDistortion node) and then refines the distortion in the camera solve. This way you don't have to calculate your lens distortion separately using the LensDistortion node.
- If you chose either **Known Lens** or **Refine Known Lens**, enter values for the **Lens Type**, **Radial Distortion**, **Distortion Center** controls and, if necessary, the **Anamorphic Squeeze**, **Asymmetric Distortion**, and **filter** controls. For more information on these controls, see "Adjusting LensDistortion Parameters" on page 683.
- After you've solved your camera and have your lens distortion values, you can also toggle the **Undistort Input** box to view your footage distorted or undistorted.

Refining the Solve

Having added your locators and created a solve using the locator point movement info, you can now refine the solve before creating a Camera node.

1. You can pick and choose which aspects you'd like to adjust in your solve. You can choose one or a combination of the following. Check:
   - **Focal Length** - to refine the focal length information of the camera.
   - **Position** - to adjust the position information of the camera.
   - **Rotation** - to adjust the rotation of the camera.
2. You can either adjust your solve in the Viewer first and then click **Refine Output** to apply your changes, or check **Live Refinement** to update the solve as you're adjusting it in the Viewer.
3. Adjust the solve by fine-tuning your locator points in the Viewer.
4. Check the **Link output** box if you want to create a link between ProjectionSolver and the Camera and Card nodes it creates. If the nodes are linked, changing values in one also changes them in the other. If you

leave the Link output box unchecked, the Camera and Card nodes are left unlinked, and you can copy and use them in another script.

5. When you're happy with your adjustments, click **Create Camera**.

   ProjectionSolver creates a Camera node.
6. You can go on to creating cards and adjusting the camera created by ProjectionSolver on the **Output** tab. In most cases you won't need to touch the controls on the **Output** tab however. For more information about these controls, see "Adjusting the Virtual Camera" on page 674.

Tip *If you have a nodal camera sequence, the camera position should be fixed. You can enforce this by setting a constant output position. The camera rotation can then be refined by selecting the* ***Rotation*** *option and* ***Refine Output*** *to get the best solve for the fixed position.*

Creating Cards

ProjectionSolver also lets you create Cards in your 3D scene to correspond with each of the camera positions. This helps you visualise the scene at a glance, without having to scrub back and forth on the timeline.

After you've created your Camera, click **Create Cards**. ProjectionSolver creates a set of Card nodes connected to your Scene node, and you can use them in the 3D view mode to view your camera positions. In the **z** field, you can specify the z distance, or the depth value for the Card nodes.

7 RENDERING WITH PRMANRENDER

PrmanRender is a render node that works together with Pixar's PhotoRealistic RenderMan® software to give you an even better quality render result. PrmanRender is an alternative to Nuke's ScanlineRender with additional features for rendering 3D scenes.

Setting Up RenderMan and PrmanRender

In order to use the PrmanRender node, you need to have the RenderMan software installed and licensed on your machine. To do this:

1. Follow the instructions in Pixar's PhotoRealistic RenderMan installation guide to get RenderMan working on your machine. This is in most cases enough to get you going with using both RenderMan and the PrmanRender node in Nuke. Note that RenderMan® specifically needs two environment variables set in order to work with Nuke:
   - RMANTREE - This needs to point to the location of your Prman distribution.
   - Depending on which operating system you're using either DYLD_LIBRARY_PATH (on Mac), LD_LIBRARY_PATH (on Linux) or PATH (on Windows). This environment variable needs to point to "%RMANTREE%/lib". For more information on setting environment variables, see "Environment Variables" on page 611. If you launch Nuke on Mac OS X from the starup icon rather than a terminal shell, be sure to add these variables to your environment.plist (see page 612).
2. To make sure your PrmanRender node is working in Nuke, start Nuke from the terminal (for more information, see "Launching Nuke" on page 33).
3. Create a PrmanRender node in Nuke (**3D > RenderMan > PrmanRender**) and connect it to your nodes. You can try the following combination of nodes as a test: Checkerboard > Sphere > PrmanRender > Viewer.

Using The PrmanRender Node

The PrmanRender node can render nearly everything that you previously used ScanlineRender for, but with PrmanRender, you have control over aspects like shadows and reflections in your render result.

Connect the PrmanRender node to your Scene node, Camera node and any optional inputs, in the same way you would connect ScanlineRender. For more information about connecting a ScanlineRender node, see "Setting Up a Scene" on page 427.

On the **PrmanRender** tab, you can select which aspects you'd like to render out by checking **shadows**, **reflections**, **refractions** or **dof** (depth of field). For more information, see "Adjusting Shadows, Reflections, Refractions and Depth of Field" on page 707.

You can select your projection mode in the **projection mode** dropdown:

- **Perspective** - objects in front of the camera have the illusion of depth defined by the camera's focal-length and aperture.
- **Orthographic** - objects are viewed using a parallel projection.

If necessary, also adjust:

- **overscan** - to set how many pixels are rendered over the right/left and top/bottom of the frame, if a subsequent operation requests this.
- **ambient** - to add global ambient lighting.

Adjusting Render Quality

On the **Sampling** tab, you can adjust controls that affect your render quality. Adjust:

- **ray trace max depth** - to set the maximum depth of the view rays PrmanRender uses to trace your scene and make render calculations.
- **pixel samples** - to set the number of samples to render per pixel. Having more samples increases your render quality, but also increases render time.
- **filter** - to choose a texture sampling filtering algorithm. For more information on the different options, see "Choosing a Filtering Algorithm" on page 227.
- **antialiasing filter** - to choose an antialiasing filter, **box**, **triangle**, **catmull-rom**, **sinc**, **gaussian**, **mitchell**, **separable-catmull-rom** or **blackman-harris**.
- **antialiasing filter size** - to choose the size of the antialiasing filter.
- **shading rate** - to set the shading calculation for primitives. This value, along with pixel samples, directly affects your rendering time and the final quality of your results. A small shading rate value means your render will take more time, but the quality will be very high. A large value on the other hand means your render is faster, but the final quality will not be as good.

Adjusting Shadows, Reflections, Refractions and Depth of Field

On the **PrmanRender** tab, you can select if you want to render shadows, reflections, refractions and depth of field, or all of them into your result. All these effects are calculated using a retracing method that is based on drawing rays from the camera to the object. Check the box for:

- **shadows** - to add shadows to your render. You can adjust the parameters for your shadows on your Light node's **Shadows** tab. For example, if you adjust the **sample width**, the shadows will be softer. For more information, see "To adjust the Shadows controls" on page 476.
- **reflections** - to add reflections to your render. You can adjust the parameters for your reflections on your Reflection node's control panel. See "Using the Reflection Node" on page 709.
- **refractions** - to add refractions to your render. You can adjust the parameters for your refractions on your Refraction node's control panel. See "Using the Refraction Node" on page 709.
- **dof** - to add depth of field to your render.

Adjusting Motion Blur Parameters

On the **Sampling** tab, you can adjust controls that affect motion blur. Adjust:

- **motion blur samples** - to set the number of samples to render per pixel when motion blurring.
- **shutter** - Enter the number of frames the shutter stays open when motion blurring. For example, a value of 0.5 would correspond to half a frame. Increasing the value produces more blur, and decreasing the value less.
- **shutter offset** - Select when the shutter opens and closes in relation to the current frame value when motion blurring:
    - **centered** - to center the shutter around the current frame. For example, if you set the shutter value to 1 and your current frame is 30, the shutter will stay open from frame 29.5 to 30.5.
    - **start** - to open the shutter at the current frame. For example, if you set the shutter value to 1 and your current frame is 30, the shutter will stay open from frame 30 to 31.
    - **end** - to close the shutter at the current frame. For example, if you set the shutter value to 1 and your current frame is 30, the shutter will stay open from frame 29 to 30.
    - **custom** - to open the shutter at the time you specify. In the field next to the pulldown menu, enter a value (in frames) you want to add to the current frame. To open the shutter before the current frame, enter a negative value. For example, a value of -0.5 would open the shutter half a frame before the current frame.
- **randomize time** - adjust this to add randomness to the distribution of samples in time so they don't produce regularly spaced images. The larger the value, the larger the time difference between the samples.
- **shutter opening** - to set the shutter opening behavior. Choose:
    - **none** - to use the default shutter opening method, resulting in instantaneous timing to open and close.

  - **linear** - to set the shutter to open and close in linear intervals.
  - **bezier** - to set the shutter to open and close more gradually, according to a Bezier curve.

Arbitrary Output Variables Parameters

On the **AOV** tab you can choose which channels are affected by motion vectors and output vectors.

- Check **motion blur vectors** if you want to render motion vectors.
- Adjust **motion vector channels** to choose which channels you want the motion vectors to be output to.
- Check **output vectors** if you want to render output vectors.
- Select which channels to apply surface points and surface normals from the **surface point** and **surface normal** dropdowns.

Using the Reflection Node

Reflection is the familiar physical phenomenon where an image of an object is cast back from a particular kind of surface, such as glass or water. Using PrmanRender node, you can replicate this effect in your render result of 3D objects, and using the Reflection node, you can adjust the controls for creating the reflection effect. PrmanRender uses raytracing to create this effect and you can use the Reflection node to adjust the result. Without the PrmanRender node and RenderMan software though, the Reflection node has no effect.

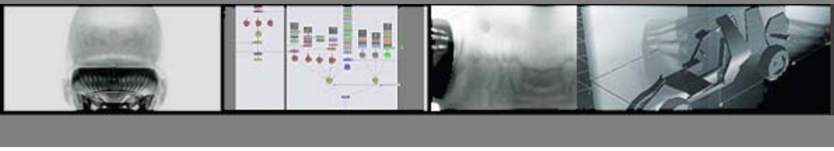
1. Create a Reflection node by clicking **3D** > **Shader** > **RenderMan** > **Reflection.**
2. Connect the node to the PrmanRender node and set the controls to adjust your reflection:
   - **reflection color** - sets the color of the reflection.
   - **value** - sets the intensity of the reflection.

Using the Refraction Node

Refraction is the familiar physical phenomenon of light traveling differently through different materials and thus reflecting differently off objects behind that material. For example, if you have a glass of water with a straw in it, the part of the straw that's not in water appears to be in a different angle to the part which is in the water. This is due to water bending the light waves. PrmanRender uses raytracing to create this effect and you can use the Refraction node to adjust the result. Without the PrmanRender node and RenderMan software though, the Refraction node has no effect.

1. Create a **Refraction** node by clicking **3D** > **Shader** > **RenderMan** > **Refraction.**

2. Connect the node to the PrmanRender node and set the controls to adjust your refraction:
   - **refraction index** - slide to change the type of refraction.
   - **value** - sets the intensity of the refraction.



TUTORIALS

Welcome to the Tutorials! If you've reviewed the Reference chapters— which we highly recommend—you already know something about Nuke. These tutorials show how to pull everything together through a series of practical examples. We will add further tutorials as time goes on. They will appear on our web site between releases of Nuke.

The Projects

- **Tutorial 1: Compositing Basics**. Explains the Nuke user interface, project workflow, and basic compositing tasks.
- **Tutorial 2: Tracking, Stabilising, and Matchmoving**. Demonstrates how to track image patterns, stabilize footage, lock down images for clean plates and matchmove.
- **Tutorial 3: Keying and Mattes**. Shows how to pull mattes with standard keying tools and Nuke's own image-based keyer.
- **Tutorial 4: 3D Integration**. Shows how you can use Nuke's 3D workspace to help your 2D compositing.

Installing the Project Files

Before you continue, download the tutorial project files from The Foundry website and move them to a directory you'll create, called "Nuke_Tutorials". It's up to you where you put your tutorial files, but here's our recommendations below depending on your operating system. Whatever you do, you'll need to remember where you put these files.

Tip *If you're using a Mac or Linux system, log in under administrator privileges to avoid issues with permissions when installing the files.*

To create the tutorial directory (Windows):

1. On the Windows desktop, double-click the **My Computer** icon to open a file browser.
2. Double-click on **Local Disk (C:)** and open the **C:\Documents and Settings\All Users\Application Data** folder.
3. Click the right-mouse button over the displayed directory and choose **New > Folder** from the pop-up menu.
4. Name the folder: **Nuke_Tutorials**.

To create the tutorial directory (Mac OS or Linux):

1. Open a shell or terminal window.

2. At the command line, enter **mkdir ~/Nuke_Tutorials** to create the tutorial directory under your user or "home" directory.

To download and install the project files:

1. Open an internet browser and go to: *http://www.thefoundry.co.uk.*
2. Navigate to the Nuke product page from the Products menu at the top of the home page.
3. Click on the Tutorials & Example Images link on the right hand side. You can also access this page by selecting **Help** > **Tutorials** in Nuke.
4. Click the links to download the project files to your local computer.
5. Extract the downloaded files and move (or copy) them to the Nuke_Tutorials directory you created earlier.

You're now ready to start the first tutorial with Nuke.

TUTORIAL 1: COMPOSITING BASICS

Hello! This tutorial is your introduction to Nuke, where you'll create a simple composite and breeze through most of the windows, on-screen controls, and other user interface items.

We've heard rumors that many people would rather march through icy rain than review an introductory tutorial on digital compositing. Certainly, that's not you. When you finish this lesson you'll have a good understanding of the Nuke workflow and should feel confident about approaching the other tutorials.

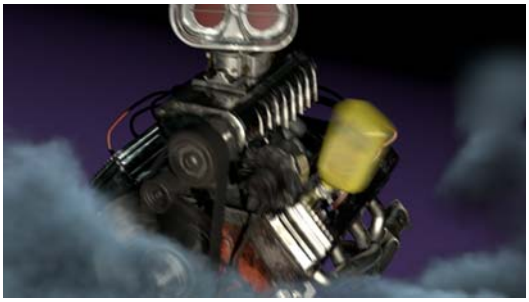


Figure 1.1: Your first composite in Nuke.

Before you get into the project, we have some administrative things to do—such as defining a few application preferences and project settings. We know this sort of thing is not terribly exciting, but it is terribly important, so please be patient and we'll get through it as quickly as possible.

Note *If you haven't already downloaded and moved the tutorial project files to the Nuke_Tutorials directory you created, turn to "Installing the Project Files" on page 711 for instructions.*

Starting Nuke

The Nuke icon may appear on your desktop. If so, double-click it to launch the application. Otherwise, start Nuke with one of the methods described below, assuming you have installed Nuke to the default location.

To launch Nuke under Windows:

- From the **Start** menu, choose **All Programs > The Foundry**, and then select Nuke6.2v2.

To launch Nuke under Mac OS X:

- Open the /**Applications/Nuke**6.2v2**/** folder and double-click the **Nuke**6.2v2 icon.

To launch Nuke under Linux:

- Open the **/usr/local/Nuke**6.2v2**/** folder and double-click the **Nuke**6.2 icon.

Tip *If you're operating under Linux, you can also launch Nuke from the command line of a terminal. Simply navigate to the Nuke directory and enter the name of the nuke application.*

A clean copy of the main Nuke window appears. Divider lines organize the window into different panes. Each pane has one or more pages of content, separated by tabs at the top of the pane. The Toolbar appears at the left edge of the main window.

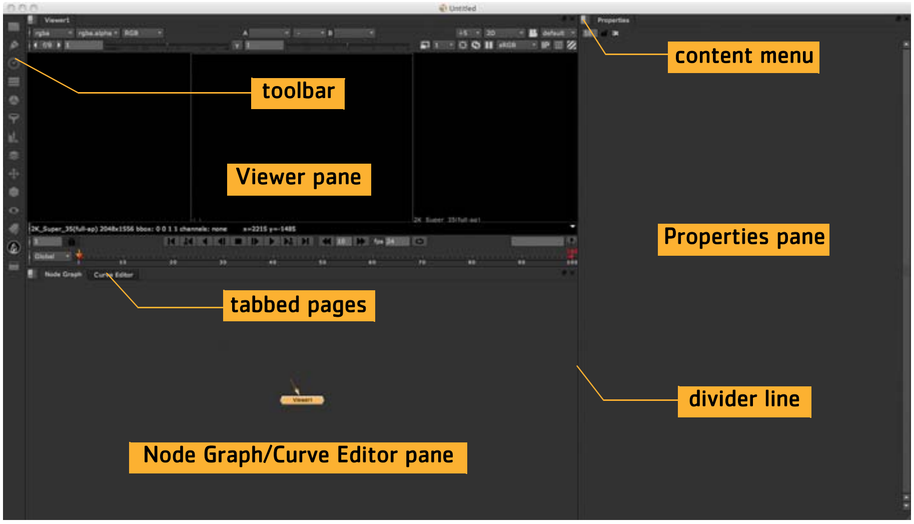


Figure 1.2: The main workspace.

By default, the panes are setup to display the Viewer, the Node Graph/ Curve Editor, and Properties. You'll create the script for this project inside the Node Graph page on the Node Graph/Curve Editor pane. We'll talk about each of these on-screen controls when you need them for the project.

Using the Toolbar

The Toolbar includes the options you can use to build your project, such as importing images, layering images, drawing shapes and masks, applying color correction, and so on. Each Toolbar icon displays a menu of *operators* or *nodes* that you can select. Roll the mouse pointer over the Toolbar and you'll see pop-up tool tips that identify each icon.

Using the Menus

The Nuke menu bar appears at the top of your screen, outside the main window. This menu begins with the options "File," "Edit," "Layout," and so on. When instructed to do so, make selections from the menu bar, or click the right mouse button to choose from a pop-up version of the menu bar.

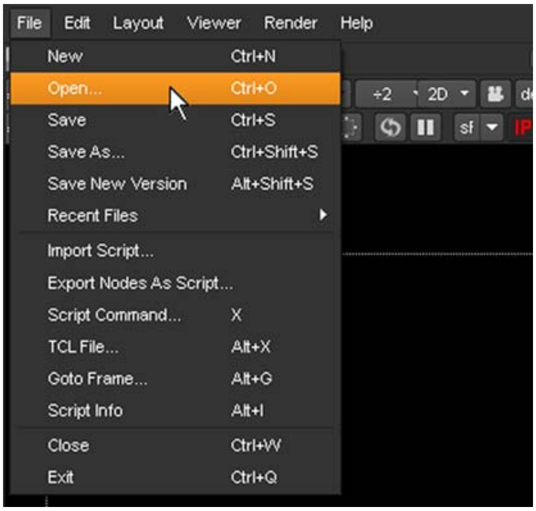

File
Edit
Layout
Viewer
Render
Help
New Ctrl+N
Open... Ctrl+O
Save Ctrl+S
Save As... Ctrl+Shift+S
Save New Version Alt+Shift+S
Recent Files
Import Script...
Export Nodes As Script...
Script Command... X
TCL File... Alt+X
Goto Frame... Alt+G
Script Info Alt+I
Close Ctrl+W
Exit Ctrl+Q
÷2
2D
sf
IP

Figure 1.3: The Nuke menu bar.

The "right-click" menu is highly contextual. Its options change according to the location of the mouse pointer. Right-click over the Node Graph, for example, and you'll see the options from the menu bar and the nodes you can insert from the Toolbar. Right-click over the Viewer pane and you'll see a menu of Viewer options.

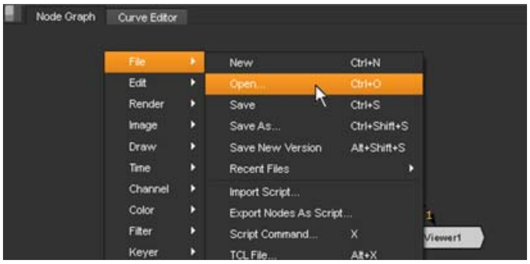

Node Graph
Curve Editor
File
Edit
Render
Image
Draw
Time
Channel
Color
Filter
Keyer
New Ctrl+N
Open... Ctrl+O
Save Ctrl+S
Save As... Ctrl+Shift+S
Save New Version Alt+Shift+S
Recent Files
Import Script...
Export Nodes As Script...
Script Command... X
TCL File... Alt+X
1
Viewer1

Figure 1.4: The "right-click" menu.

Try the right-click menu when you can't find appropriate controls or menu options. Many features are hidden in the pop-up menu until you're in the situation where you need to use them.

Note *Nuke's menu bar, at the top of the screen, is organized a little differently between the operating systems, but the right-click menu contains the same options, regardless of the system you're using to run Nuke.*

Customizing Your Layout

Nuke gives you several options for customizing the window layout. It's time for you to claim your copy of Nuke and make it your own! You don't need to customize the layout for this lesson, but why not try it now for your own personal amusement? Here are some things you can do to reorganize the window layout:

- Drag a divider line between panes to change the size of the panes.
- To divide a pane, click on the content menu (the checkered box at the upper-left corner of each pane), and choose **Split Vertical** or **Split Horizontal**.
- To add a new tabbed page to a pane, click on the content menu and choose one of content options, such as **New Viewer** or **Curve Editor**.
- Click on the "x" inside a tab to discard a tabbed page.
- To move a tabbed page, drag the tab to another pane inside the main window.
- To tear-off a page as a floating window, drag the tab outside the borders of the main window, or simply **Ctrl**+click (Mac users **Cmd**+click) on the tab name.
- Drag a floating window into a pane, inside the main window, to convert it to a tabbed page.
- From the menu bar, choose **Layout > Save Layout** *x* to save the current layout. Choose **Layout > Restore Layout** *x* to apply a previously-saved layout.
- To select a predefined color scheme, click the right mouse button and choose **Edit > Preferences**. Then click the **Choose a Preset** button and select a color scheme.
- Define other appearance options, such as window colors and fonts, by changing the settings under **Edit > Preferences >** the **Node Graph** tab.

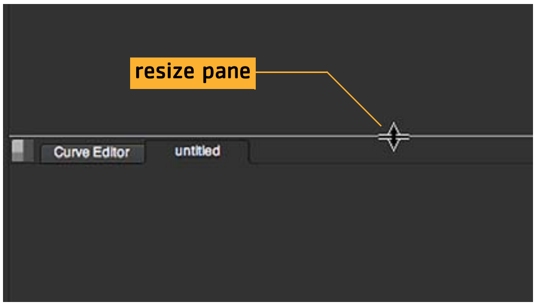

resize pane
Curve Editor
untitled

Figure 1.5: Resizing a pane.

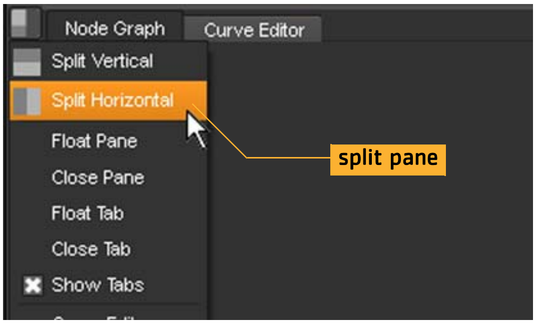

Node Graph
Curve Editor
Split Vertical
Split Horizontal
Float Pane
Close Pane
Float Tab
Close Tab
Show Tabs
split pane

Figure 1.6: Splitting a pane.

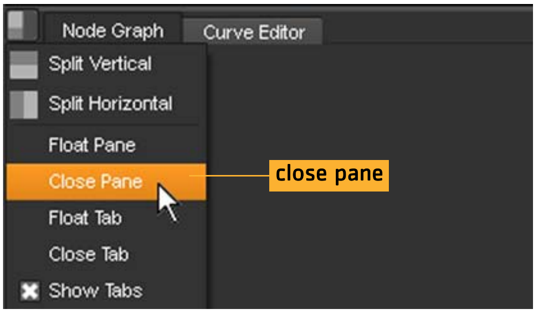

Node Graph
Curve Editor
Split Vertical
Split Horizontal
Float Pane
Close Pane
Float Tab
Close Tab
Show Tabs
close pane

Figure 1.7: Closing a pane.

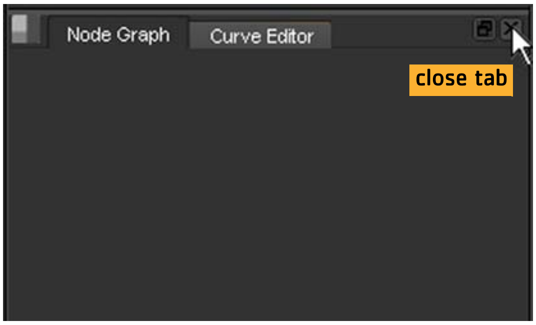

Node Graph
Curve Editor
close tab

Figure 1.8: Closing a tab.

Saving Files and File Backup

We assume you already know how to save files (Hint: choose **File > Save As**). In addition, Nuke includes an autosave feature, which helps recover project files after a system failure. Yes, we know that will *never* happen to you, but in the unlikely event that it does, you won't lose your work when you have autosave enabled.

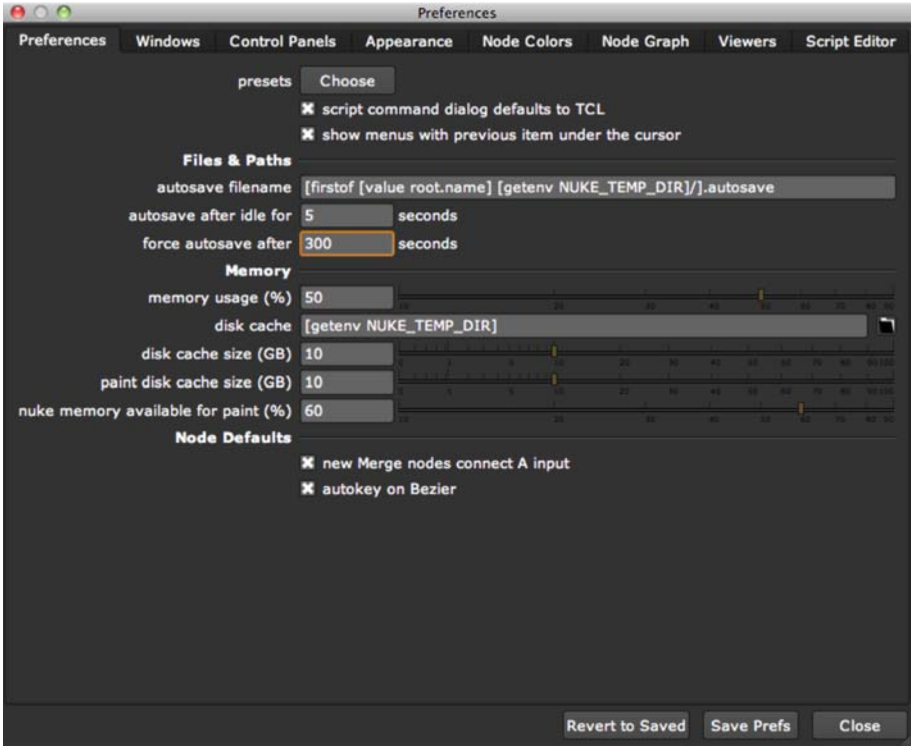
To define file/saving options:

1. Click the right mouse button over the Node Graph pane, and choose **Edit > Preferences**.

   Notice the "autosave filename" directory is set to "[value root.name].autosave." You don't need to make a change; this simply tells Nuke to store automatic backup files in the same directories as your project files.

   Now, how often would you like Nuke to generate an automatic backup while you're working? Every five minutes?

2. Change the **force autosave after** option to **300** seconds, to generate an automatic backup every five minutes.



presets Choose
☒ script command dialog defaults to TCL
☒ show menus with previous item under the cursor

Files & Paths
autosave filename [firstof [value root.name] [getenv NUKE_TEMP_DIR]/].autosave
autosave after idle for 5 seconds
force autosave after 300 seconds

Memory
memory usage (%) 50
disk cache [getenv NUKE_TEMP_DIR]
disk cache size (GB) 10
paint disk cache size (GB) 10
nuke memory available for paint (%) 60

Node Defaults
☒ new Merge nodes connect A input
☒ autokey on Bezier

3. Click **Save Prefs** to keep the changes and then **Close** to return to the main window.

If you close this dialog box without clicking the Save button, then the changes will affect only the current session of Nuke.

Recovering backup files

You may ask, "How do I recover a backup file in the event of a system or power failure?" Good question! When you relaunch Nuke, you'll see a message that asks if you want to recover the .autosave file for the project that was last open. Click **Yes** and Nuke opens the backup file.

Tip *The .autosave files can still be useful, even when you properly exit Nuke, because they are not deleted from the directory. You can, for example, rename an .autosave file to create an archive of the previous version of your project file.*

Sometimes you will see the recovery message even though you have not experienced a system failure. This happens when you exit Nuke without saving the changes to a project file, and Nuke recognizes that the time stamp on the .autosave file is later than the Nuke project file you're trying to open. In this case, you decide which version of the project file you want to open.

Turning off automatic backup

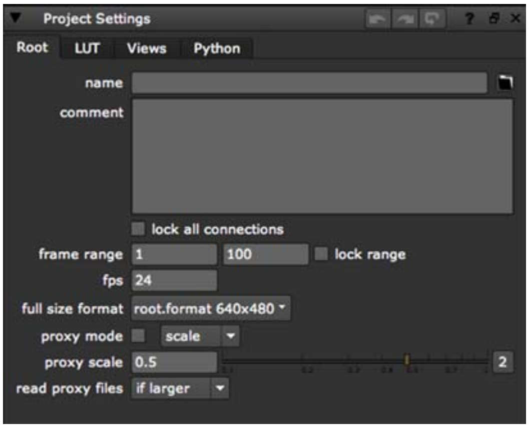
Okay. You're reading this, so we assume you're a freewheeling rebel who possibly enjoys the risk of losing your work. It's an adrenaline thing. Or perhaps you prefer to do everything yourself, manually, and you have a secret obsession for saving your files, often. Whatever the reason, you can disable the autosave features by setting the intervals for both "autosave idle" and "force autosave" to zero seconds. That's it. That's all you need to do. Good luck.

Setting Up the Project

When you start a new project, you need to define project settings for length or frame range, the number of frames per second for playback, and the output format. These options appear on the Project Settings dialog box.

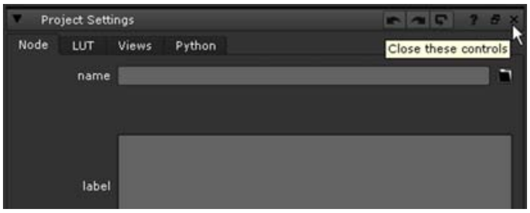
To setup your project:

1. Click the right mouse button over the Node Graph, and then choose **Edit > Project Settings** from the pop-up menu.


Project Settings
Root
LUT
Views
Python
name
comment
lock all connections
lock range

frame range 1 100
fps 24
full size format root.format 640x480
proxy mode scale
proxy scale 0.5 2
read proxy files if larger

2. In the **frame range** fields, enter a range of **1** to **28**. This will be the length of the shot we create for the project.
3. Enter **24** as the frames per second (fps).
4. Click the **full size format** list and choose **PC_Video 640 x 480**.
5. Close the **Settings** control panel.


Project Settings
Node
LUT
Views
Python
Close these controls
name
label

Note *On the project Settings control panel, the LUT tab includes options that ensure color integrity for your display and output devices. You don't need to change the LUT for these tutorials, but we recommend that you research and set these options for your own projects.*

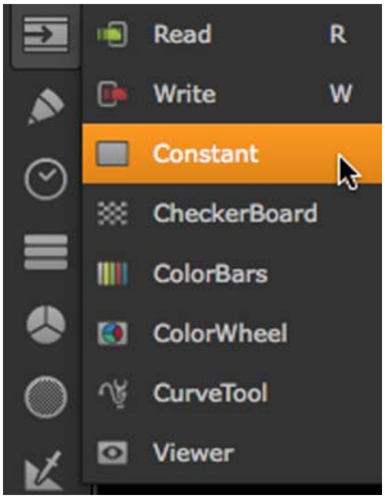
Until now, everything you've done is standard procedure for a new project. You used the menu bar to access several features during the setup process, and now you'll use the Nuke toolbar to insert nodes and create a compositing tree.

Working with Nodes

A *node* is simply one of the building blocks for the list of operations you want to complete. A *node tree* is a diagram that shows the order in which the operations will be performed. Do the following to add a few nodes and start your node tree. The result will create the background for the project.

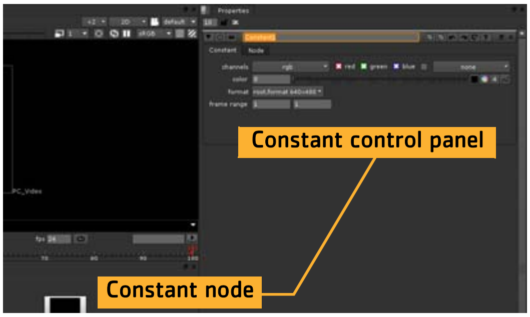
To insert nodes:

1. On the Toolbar, click the first icon to display a menu for nodes that are in the **Images** category.

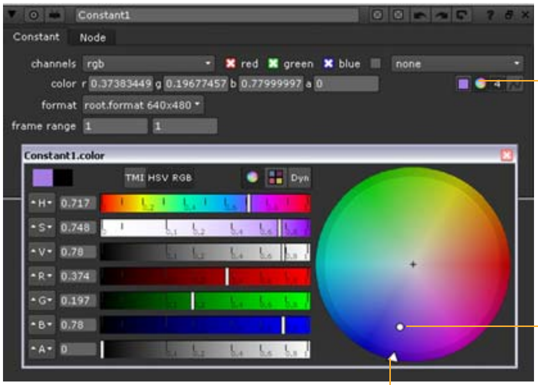

Read R
Write W
Constant
CheckerBoard
ColorBars
ColorWheel
CurveTool
Viewer

2. Select **Constant** from the menu to insert this node into the Node Graph pane.

   When you insert a new node, its control panel also appears with parameters that let you define what the node will produce. In this case, the Constant node creates a solid color backdrop.


Constant control panel
Constant node

3. In the Constant control panel, click on the color wheel to open the Color Picker.



channels rgb ☒ red ☒ green ☒ blue ☐ none

color r 0.37383449 g 0.19677457 b 0.77999997 a 0

format root.format 640x480

frame range 1 1

H 0.717

S 0.748

V 0.78

R 0.374

G 0.197

B 0.78

A 0

Color wheel icon

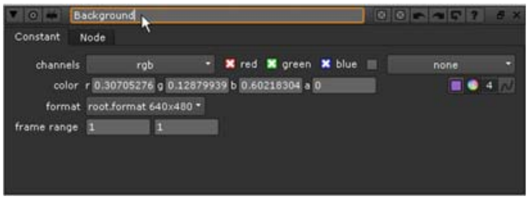
Drag inside the wheel to select a value.

Drag along the ring to select a color.

4. Drag the color sliders and the cursor inside the wheel to choose a light color, something appropriate for the "horizon" of the composite background. Then, close the color wheel window.

   At this point, you should probably rename "Constant" to something more descriptive.

5. Inside the control panel, click on the **Constant** name. You can now edit the name, so type **Background** and press **Enter**.



channels rgb ☒ red ☒ green ☒ blue ☐ none

color r 0.30705276 g 0.12879939 b 0.60218304 a 0

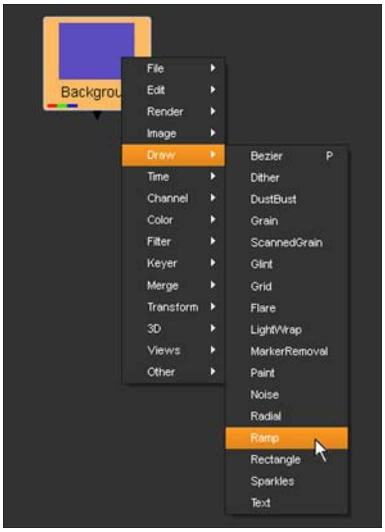
format root.format 640x480

frame range 1 1

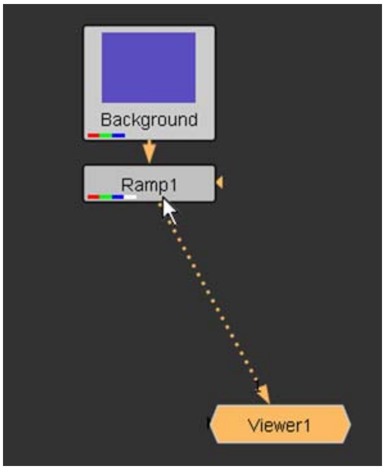
   From here onward, we'll call this node the "Background" node.

6. Close the control panel for the Background node. When you need to reopen it, just double-click the node and the control panel will reappear.

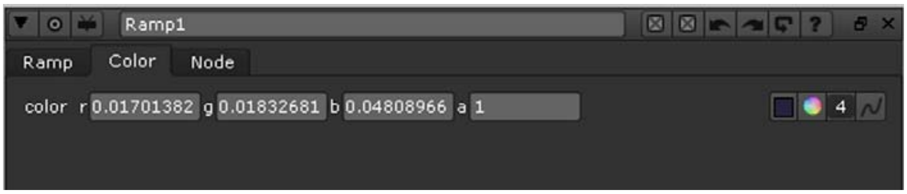
7. Click on the **Background** node to select it. Then, click the right mouse button and choose **Draw > Ramp** from the pop-up menu.


Backgrou
File
Edit
Render
Image
Draw
Time
Channel
Color
Filter
Keyer
Merge
Transform
3D
Views
Other
Bezier P
Dither
DustBust
Grain
ScannedGrain
Glint
Grid
Flare
LightWrap
MarkerRemoval
Paint
Noise
Radial
Ramp
Rectangle
Sparkles
Text

8. Drag the tail of the arrow from the **Viewer1** node to the center of the **Ramp1** node. You'll see the output of the Background node and the ramp controls displayed in the Viewer window.

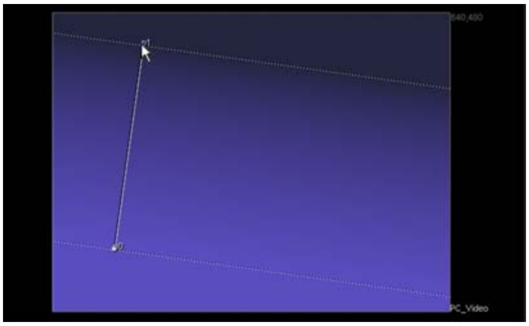

Background
Ramp1
Viewer1

9. Click the **Color** tab inside the control panel for **Ramp1**. Then choose a dark color that blends well with the color you selected for the Background node.


Ramp1
Ramp
Color
Node
4

color r 0.01701382 g 0.01832681 b 0.04808966 a 1

10. Click the Ramp tab in the control panel to reactivate the overlay controls. Then, drag the **p0** and **p1** control points to adjust the spread and angle of the ramp over the background.


640,400
p1
p0
PC_Video

11. When you're happy with the results, close the Ramp1 control panel to remove the overlay.

Connection Tips

Most nodes have input and output connectors that are used to establish the order in which the operations will be calculated.

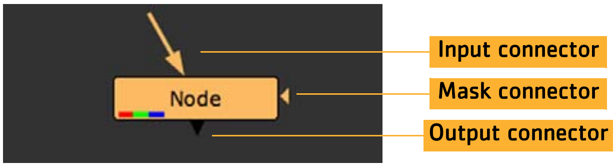
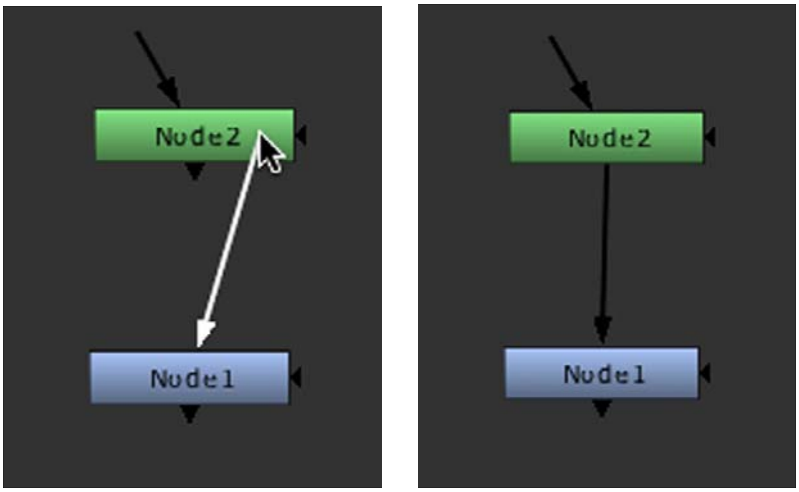

Node
Input connector
Mask connector
Output connector

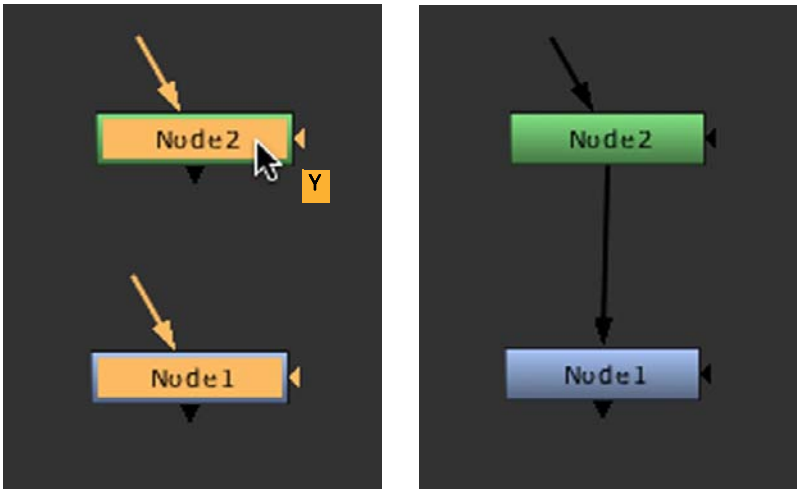
Figure 1.9: Connectors on a node.

Try the following to connect nodes after you insert them into the Node Graph:

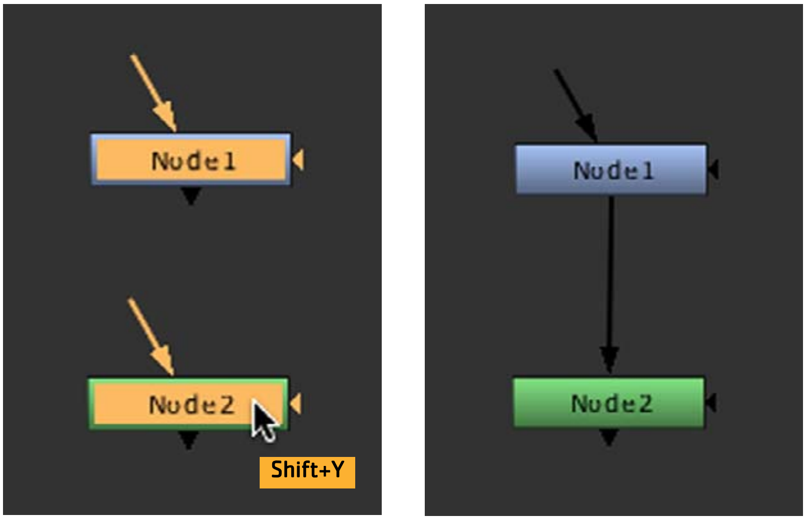
- Drag an input or an output connector onto another node to establish a connection.


Node2
Node1
Node2
Node1

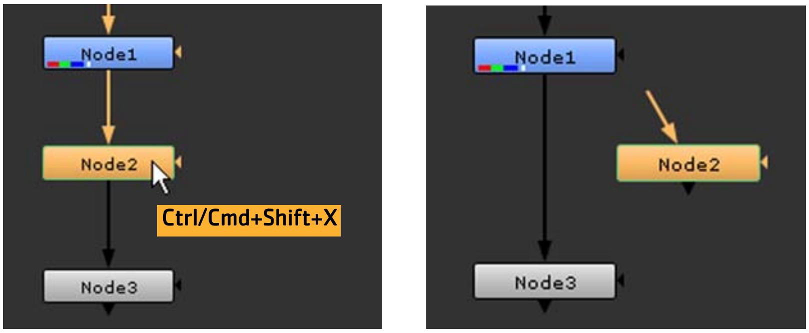
- Select a node, press the **Shift** key and select a second node. Then press **Y** to connect the first node to the output of the second node.


Node2
Y
Node1
Node2
Node1

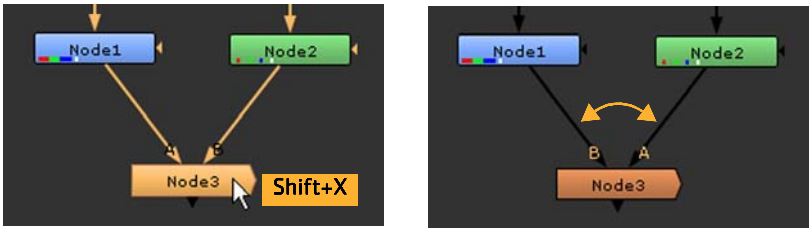
- Select a node, press the **Shift** key and select a second node. Then press **Shift+Y** to connect the second node to the output of the first node.


Node1
Node2
Shift+Y
Node1
Node2

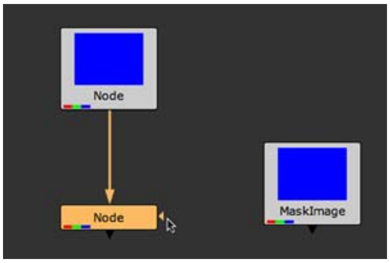
- Select a node and press **Ctrl/Cmd+Shift+X** to extract the selected node from the tree.

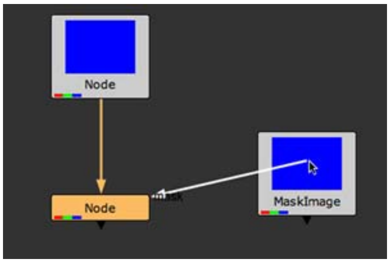

Node1
Node2
Ctrl/Cmd+Shift+X
Node3
Node1
Node2
Node3

- For nodes that have two inputs, select the node and press **Shift+X** to swap the A/B inputs.


Node1
Node2
A
B
Node3
Shift+X
Node1
Node2
B
A
Node3

- Drag the mask connector to the node that provides the image you want to use as the mask for the selected node.


Node
Node
MaskImage

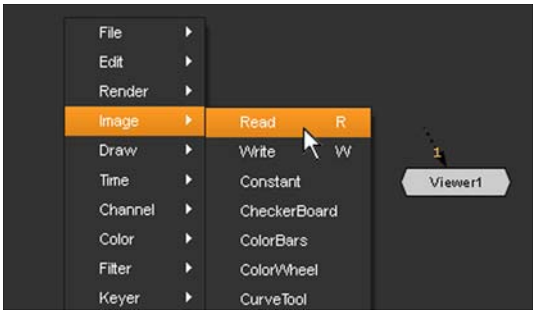

Node
Node
MaskImage

Importing Image Sequences

For this project, you need to import a few image sequences for the foreground elements and a background plate.

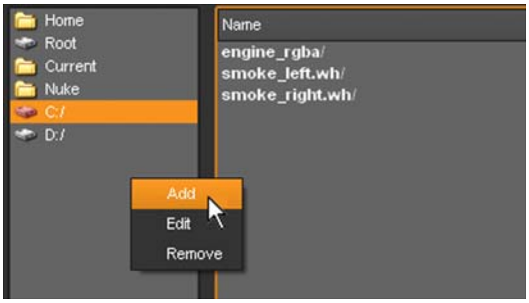
To read the images:

1. Click on a blank space in the Node Graph. This ensures none of the nodes are selected.
2. Click the right mouse button over the Node Graph and choose **Image > Read** from the pop-up menu.

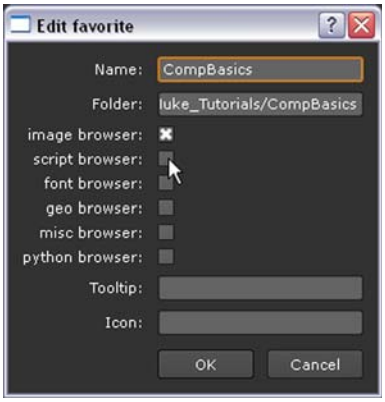

File
Edit
Render
Image
Draw
Time
Channel
Color
Filter
Keyer
Read R
Write W
Constant
CheckerBoard
ColorBars
ColorWheel
CurveTool
1
Viewer1

A file browser appears. This where you select the image file you want to import. When you browse through your directories from this window, Nuke will display sequentially-numbered files as one item in the directory.

3. Browse to the **Nuke_Tutorials/CompBasics/** directory.
4. Add a bookmark to this directory. Right-click over the list, on the left side of the file browser window, and choose **Add** from the pop-up menu.


Home
Root
Current
Nuke
C:/
D:/
Name
engine_rgba/
smoke_left.wh/
smoke_right.wh/
Add
Edit
Remove

5. Type a name for the bookmark or keep the default, which is the directory name. Then click **OK**.


Edit favorite

Name: CompBasics
Folder: luke_Tutorials/CompBasics
image browser: ☒
script browser: ☐
font browser: ☐
geo browser: ☐
misc browser: ☐
python browser: ☐
Tooltip:
Icon:

OK
Cancel

6. Open the **engine_rgba** directory, select the **engine.v01.####.exr** image sequence, and click **Open**.

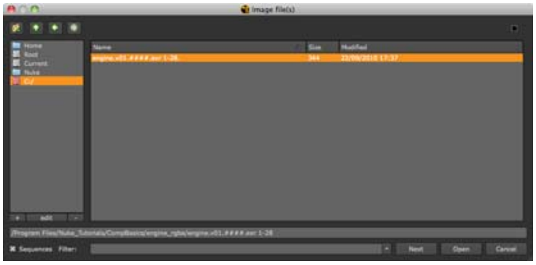
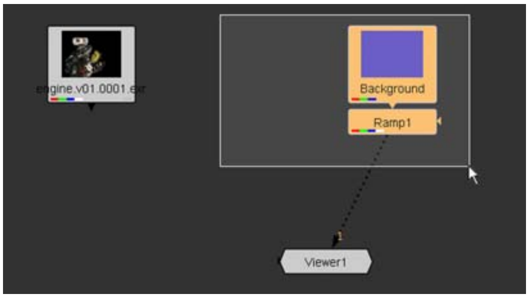


Image file(s)
Name
Size
Modified
Home
Root
Current
Nuke
C:/
edit
Sequences
Filter:
Next
Open
Cancel

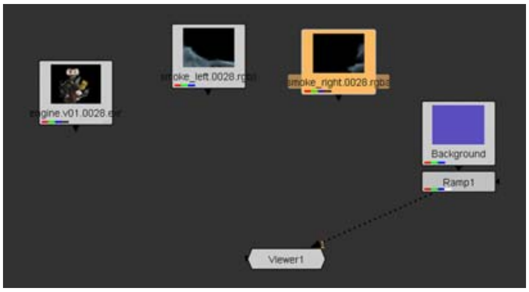
Nuke retrieves the image sequence and displays it as a thumbnail on the node. The Read control panel displays the resolution and the frame range for the image.

Note *Nuke reads images from their native format, but the Read node outputs the result using a linear color space. If necessary, you can change the Colorspace option in the Read node's control panel, or insert a* **Color > Colorspace** *node to select the color scheme you want to output or calculate.*

7. Drag a marquee (hold down the left mouse button while dragging) around the Background and Ramp nodes to select them. Then drag them to the right to make room for additional nodes.


engine.v01.0001.exr
Background
Ramp1
Viewer1

8. Choose **Image > Read** from the right-click menu to import another image sequence. Use the file browser to select the image sequence stored in **Nuke_Tutorials/CompBasics/smoke_left.wh/ smoke_left.####.rgba**.
9. Add one more Read node and retrieve the image sequence stored in **Nuke_Tutorials/CompBasics/smoke_right.wh/ smoke_right.####.rgba**


engine.v01.0028.exr
smoke_left.0028.rgba
smoke_right.0028.rgba
Background
Ramp1
Viewer1

10. Arrange the nodes, as shown above, to allow some room to create the connections for the node tree.

Navigating Inside the Windows

The node graph panel can seem very small, especially when your node tree grows. True, you already know how to resize and tear-off the windows, but sooner or later you will run out of display real estate. It's time to learn some navigation controls that will help you work in the node graph (and other windows) in Nuke. Try the following navigation controls:

Panning your view

- Windows/Linux: While pressing the **Alt** key and the left mouse button, drag the mouse pointer across the node graph.

- Mac OS X: While pressing the **Option (alt)** key and the left mouse button, drag the mouse pointer across the node graph.

  As you drag the mouse, you pan your view of the node graph.

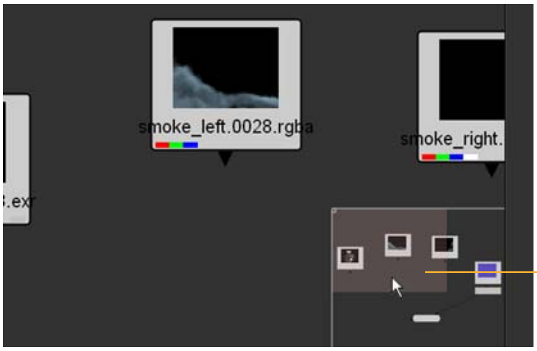
Zooming or magnifying your view

- Windows/Linux: While pressing **Alt** and the middle mouse button, drag the mouse pointer across the node graph.
- Mac OS X: While pressing **Option (alt)** and the middle mouse button, drag the mouse pointer across the node graph.

  Drag to the right and you'll zoom-in. Drag to the left and you'll zoom-out.
- Keyboard zoom-in/out. Tap the plus (**+**) key to zoom-in. Tap the minus key (**-**) to zoom-out.

Using the node graph overview

- When the node tree extends beyond the borders of the window, a navigation box appears in the lower-right corner of the node graph. Drag the shaded rectangle inside the box and you'll quickly pan to another view of the node tree.



Drag this box to pan the view of node tree

Framing the view in the window

- Press the letter **F** on your keyboard to fit the entire contents of the node tree within the borders of the node graph.

The navigation controls for the node graph also work inside the next window on our agenda, the Viewer.

Working with Viewers

The *postage stamps* on the nodes—those little pictures, often called thumbnails—show what each node passes onto the next node in the tree. Although quite lovely, they won't do for real compositing work. You need to open a Viewer window to see the full picture.

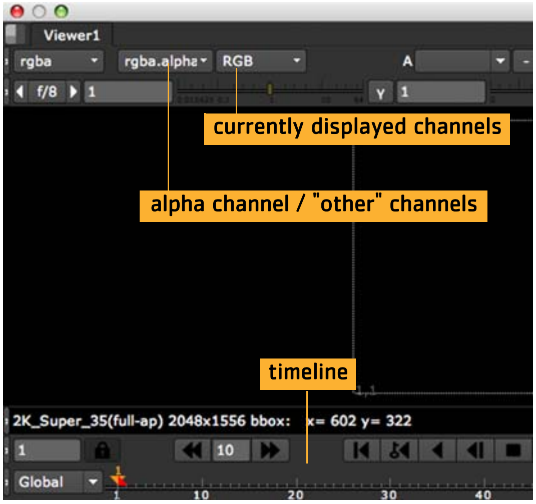

Viewer1
rgba
rgba.alpha
RGB
A
f/8
1
y
1
currently displayed channels
alpha channel / "other" channels
timeline
2K_Super_35(full-ap) 2048x1556 bbox:
x= 602 y= 322
1
10
Global
1
10
20
30
40

Figure 1.10: Controls on the left side of the Viewer.

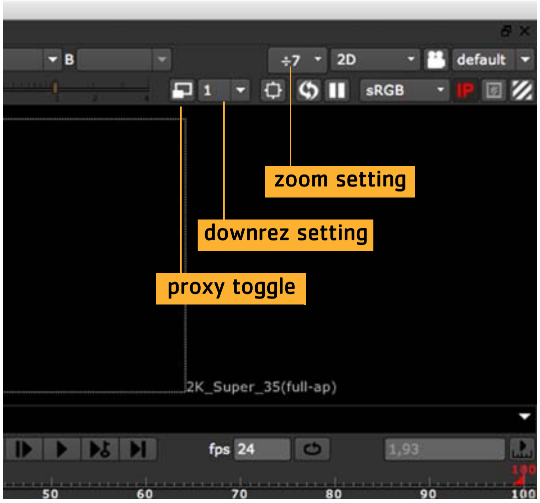

B
÷7
2D
default
1
sRGB
zoom setting
downrez setting
proxy toggle
2K_Super_35(full-ap)
fps 24
50
60
70
80
90
100

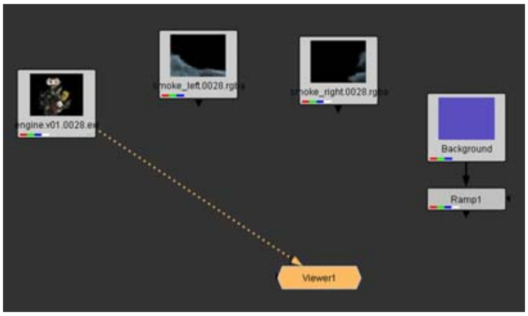
Figure 1.11: Controls on the right side of the Viewer.

You can open several Viewers at once. In addition, you have up to 10 pages, or buffers, for each Viewer window; these allow you to toggle between different views along the node tree.

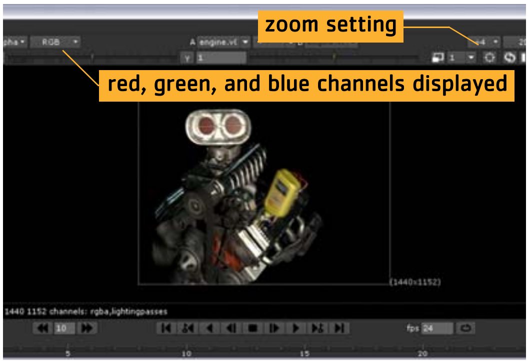
When you start Nuke, you will see a default Viewer node in the Node Graph. You can easily drag the connection arrow from a node onto the Viewer to display the node's output. You can open additional Viewers by choosing **Viewer** > **Create New Viewer** from the menu bar or by pressing **Ctrl+I**.

To display the images in a Viewer window:

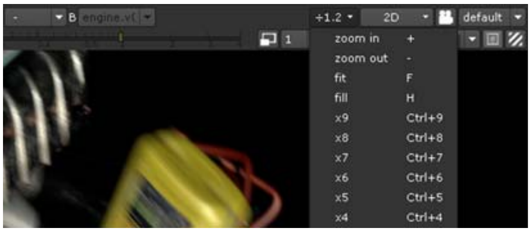
1. Drag the connector from the Viewer node onto the Read node for the **engine.v01** clip.


ngine.v01.0028.ex
moke_left.0028.rgba
smoke_right.0028.rgba
Background
Ramp1
Viewer1

Here's an alternate method: Select the **engine.v01** clip node and then press **1** to connect to the Viewer node. Nuke displays the node's output in the Viewer window.


zoom setting
red, green, and blue channels displayed
RGB
(1440x1152)
1440 1152 channels: rgba,lightingpasses
fps 24

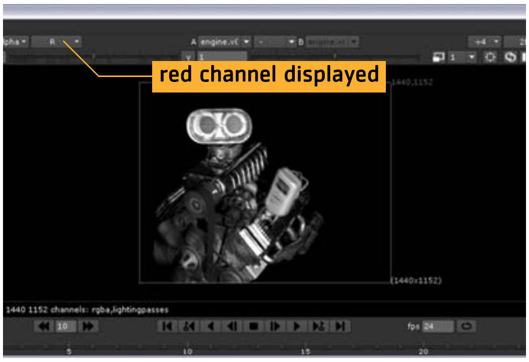
2. Press the **Alt** key (Mac users press **Option**) and the left mouse button, and drag the mouse pointer across the Viewer window to pan.
3. Press **Alt** (Mac users press **Option**) and the middle mouse button, and drag to zoom in/out. You can also use the "zoom" list at the top of the Viewer to magnify the view.


B
engine.v(
+1.2
2D
default
1
zoom in +
zoom out -
fit F
fill H
x9 Ctrl+9
x8 Ctrl+8
x7 Ctrl+7
x6 Ctrl+6
x5 Ctrl+5
x4 Ctrl+4

4. Press **F** to fit the current image into the borders of the Viewer window.

   This image has different *channels* of information you can view. The "RGB" label appears at the top because the Viewer now shows the result of the red, green, and blue channels.

5. To view individual color channels, press **R** (red), **G** (green), **B** (blue) or **A** (alpha). As you press each hotkey, the label at the top of the Viewer reflects the displayed channel.


red channel displayed

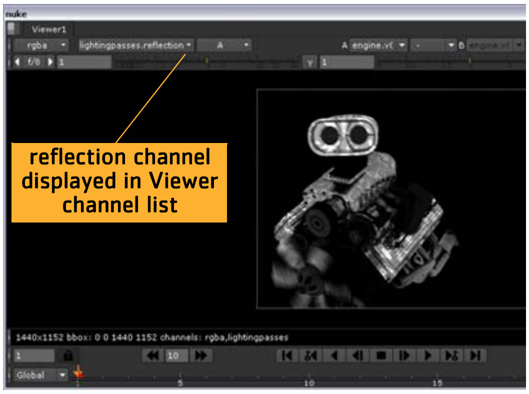
1440 1152 channels: rgba,lightingpasses

R
A engine.v(
1440,1152
(1440x1152)
fps 24
10
5
15
20

6. Press one of the channel hotkeys again to return to the "RGB" display, or choose **RGB** from the Viewer's channel list.

   In addition to the standard color channels for red, green, blue, and alpha, this image also includes channels for specular highlights, reflections, and other masks.

7. To view additional channels, press **A** to display the alpha channel, and then select the **lightingpasses.reflection** channel from the Viewer channel list.


reflection channel displayed in Viewer channel list
Viewer1
rgba
lightingpasses.reflection
1440x1152 bbox: 0 0 1440 1152 channels: rgba,lightingpasses
Global

You now see the reflection mask from the image file.

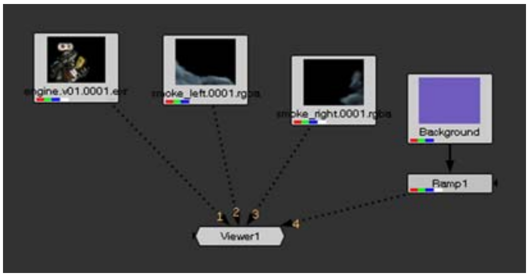
8. Select **rgba.alpha** from the Viewer channel list to reset this as the preferred channel when you press the A key.
9. Press **A** again to toggle the display and show all color channels.

To view multiple inputs:

1. Select the Read node for the **smoke_left** clip, and press **2** at the top of your keyboard or on the numeric key pad.

   This creates a second connection to the Viewer from the selected node. When the cursor is over the Viewer, you can press a number on the keyboard to pick the connection you want to view.
2. Move the mouse pointer over the Viewer and press **1** to display the **engine.v01** clip. Press **2** to display the result of the **smoke_left** node.

   In this manner, you can connect multiple images to the same Viewer and then switch between the images.
3. Select each of the other nodes and press a number to establish a connection to the Viewer.


engine.v01.0001.exr
smoke_left.0001.rgba
smoke_right.0001.rgba
Background
Ramp1
1
2
3
4
Viewer1

4. Move the mouse pointer over the Viewer and press the numbers on your keyboard to display each of the connected nodes.

As you switch between the different views, the images may appear to be the same size. However, if you look in the lower-right corner of the Viewer, you'll see the images have different resolutions.

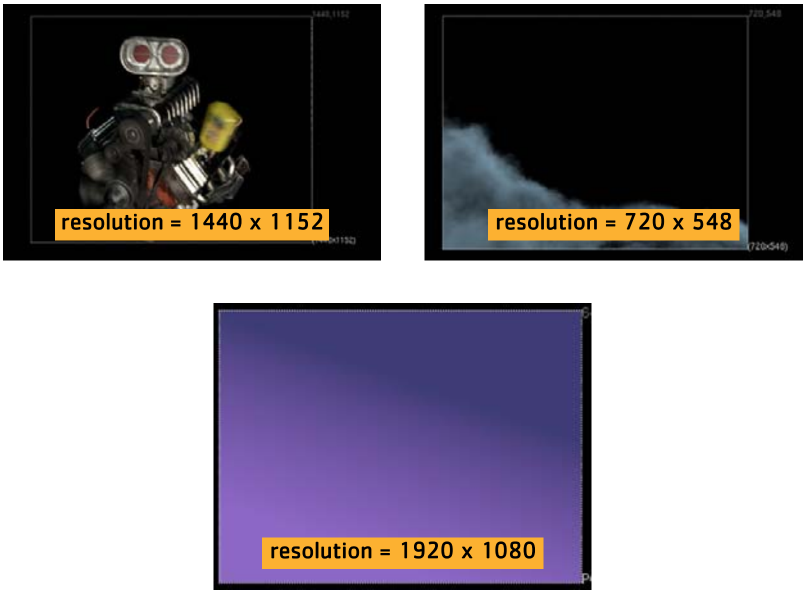


Figure 1.12: These images have different resolutions.

Nuke allows multiple resolutions in one composite, but you need to conform these images to match the project resolution. This will allow the elements to be properly aligned in the composite.

Reformatting Images

Elements created within the Nuke script, such as Background and Ramp, automatically inherit the global format and that's how you want it for this project. The imported images, however, do not conform to the project settings and must be reformatted.

To conform images to the project format:

1. Click the Read node for the **engine.v01** clip to select it.
2. Click the right mouse button and choose **Transform > Reformat** from the pop-up menu.
3. Repeat steps 1 and 2 for all the Read nodes in the node graph.
4. Move the mouse pointer over the Viewer, and press the keyboard numbers (**1**, **2**, and **3**) to switch between the connected images.

   Each image should now conform to the project format.

If you change the delivery format in the project settings, then all elements set to "root.format" will also change to the new project settings. If you neglect to reformat images when you read them into the project, the images will retain their original format, independent of the project settings.

Using Proxies and "Down-res"

Proxies are low-resolution versions of the final image you intend to create. For many compositing tasks, the low-res version can help you work faster. Then, when you're ready to create the final output, switch proxy mode off and return to the full-res version.

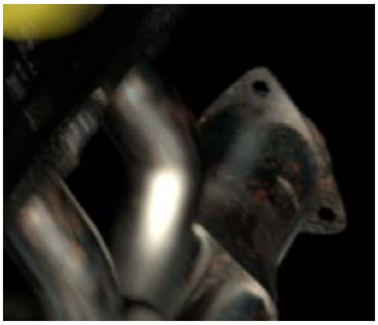


Figure 1.13: Full resolution.

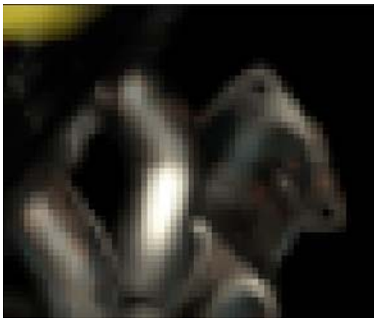


Figure 1.14: Proxy resolution.

Nuke can generate proxies on-the-fly, according to the *scale* or *format* of your images. You select the method under **Edit** > **Project Settings**.

To toggle the proxy resolution defined under Project Settings, you use the "proxy" button on your Viewer. Alternatively, you use the "down-res" button to lower the display resolution of individual Viewers. The down-res button works both in the full-res and proxy mode.

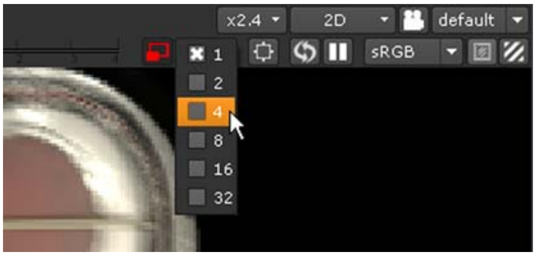
To activate proxy mode:

1. Click the right mouse button over the node graph and choose **Edit > Project Settings**.
2. Make sure the Viewer window is open.
3. Press the keystroke to toggle Proxy mode, **Ctrl+P**.

   A label inside the Viewer indicates that you are now in proxy mode.
4. Move the mouse pointer over the Viewer, and press the plus (+) key several times to zoom-in.
5. Press **Ctrl+P** a few times to toggle between hi-res and proxy mode.
6. Before you continue, press **Ctrl+P** to switch back to full resolution.

To activate "down-res":

1. Choose **4** from the "down-res" list to change the display resolution to 25% of full resolution.



   With a reduced resolution, Nuke requires less time to calculate and display your images.

2. Change the "down-res" setting back to **1**, which is 100% of the active resolution.

   If you turned off the proxy mode, you should be back to full resolution. If proxy mode is turned on, the display resolution will be 100% of the proxy resolution.

Compositing Images

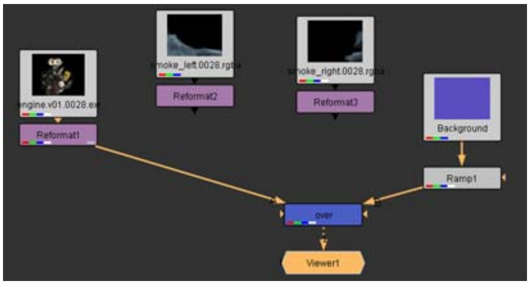
The Merge nodes create a composite with two or more images, using various compositing algorithms. In this example, we'll do a very simple "A over B" composite to layer the foreground image over the background.

You can insert a compositing node from the Toolbar or menus, but we'll show you a shortcut that bypasses both of these. The trick is the select both nodes you want to composite and then press a hotkey to assign a compositing node.

To composite two nodes:

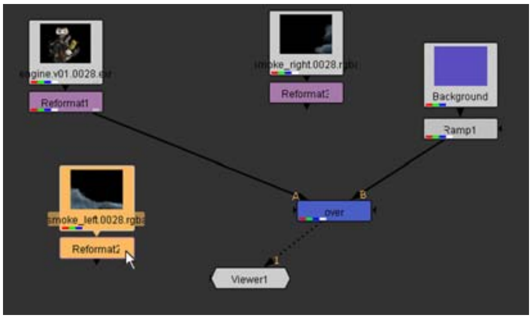
1. Select the **Reformat1** node, attached to **engine.v01**. This provides the foreground image for the first compositing operation.
2. Press the **Shift** key and select the **Ramp1** node. Both "engine.v01" and "Ramp1" nodes should be selected.
3. Press the letter **M** to insert a Merge node.

   The first node you selected is attached to the A input on the Merge node, as the foreground input. The second node you selected is attached to B, the background input. If necessary, you can swap the A and B inputs of a merge node by pressing **Shift+X**.

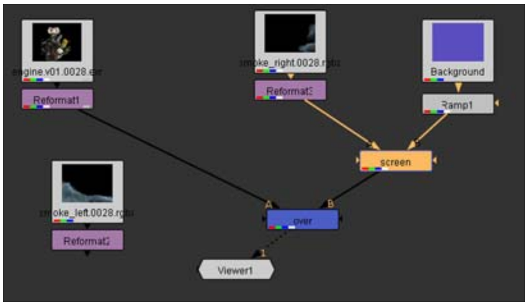

moke_left.0028.rg
noke_right.0028.rg
ngine.v01.0028.ex
Reformat1
Reformat2
Reformat3
Background
Ramp1
over
Viewer1

In the Merge node control panel, the **operation** parameter determines the compositing algorithm used to generate the result of the two inputs—the selected operation becomes the name of the node in the Node Graph.

4. Rearrange the nodes, so that the node tree looks similar to this:

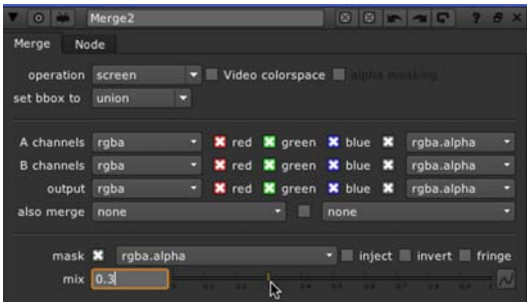

gine.v01.0028.
ke_right.0028.r
Reformat1
Reformat3
Background
Ramp1
smoke_left.0028.rgb
Reformat2
A
B
over
Viewer1

5. For the next layer, select the **Reformat3** node, attached to **smoke_right**. Then, hold down the **Shift** key and select the **Ramp1** node.
6. Press **M** to insert a Merge node and composite one image over the other. This composites the "smoke_right" image over the background.


gine.v01.0028.
ke_right.0028.r
Background
Reformat1
Reformat3
Ramp1
screen
oke_left.0028.r
Reformat2
A
B
over
Viewer1

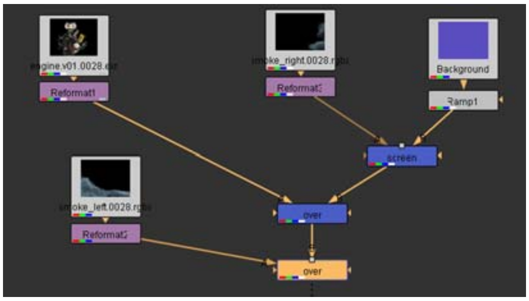
7. The default compositing algorithm, "Over," isn't exactly what we need here. In the Merge2 control panel, click on the **operation** list and select **screen**.

8. In the Merge2 control panel, drag the **mix** slider and change its value to **0.30** to reduce the amount of the image supplied by the A input.


Merge2
Merge
Node

operation: screen
☐ Video colorspace
☐ alpha masking
set bbox to: union
A channels: rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha
B channels: rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha
output: rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha
also merge: none ☐ none
mask ☒ rgba.alpha ☐ inject ☐ invert ☐ fringe
mix: 0.3

9. An additional Merge node is required. Select **Reformat2** for **smoke_left**. Hold down the **Shift** key and select the **Over** node (the first Merge node you inserted).


engine.v01.0028.exr
Reformat1
smoke_right.0028.rgb
Reformat2
Background
Ramp1
screen
smoke_left.0028.rgb
Reformat2
over
over

10. Press **M** to composite the two nodes. In the Merge3 control panel, change the **mix** slider to **0.75**.

    The result of your composite should look similar to the example shown below.

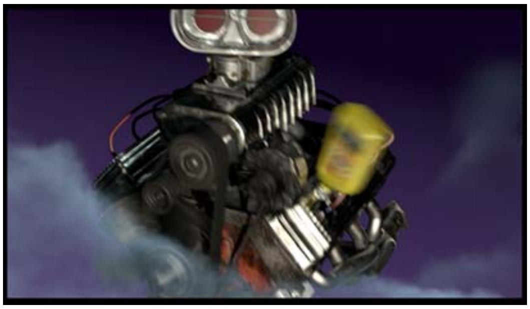
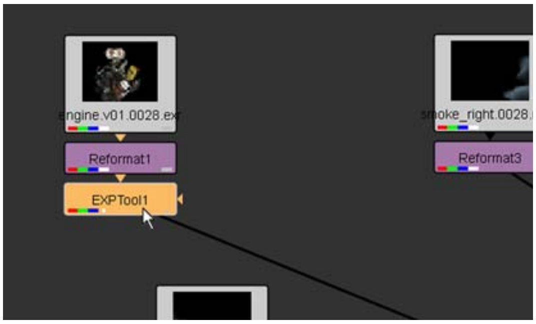


Figure 1.15: Result of the composite.

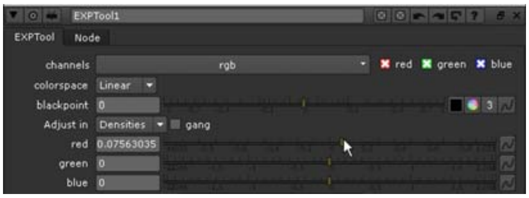
Color-Correcting Images

Color-correction and filters can help you integrate the elements for a better composite. In our example, you want to limit the correction to the foreground element only, so you'll insert a color correction node before the Merge nodes.

1. Select the **Reformat1** node. Then, right-click over the Node Graph and choose **Color > Exposure**. This inserts the EXPTool1 node.



2. Suppose you want to adjust the value of the red color channel. Move the mouse pointer over the Viewer window and press **R** to display the red channel.
3. In the EXPTool1 control panel, uncheck the box for **gang** sliders. This will allow you to adjust individual color channels.



channels rgb ☒ red ☒ green ☒ blue
colorspace Linear
blackpoint 0
Adjust in Densities ☐ gang
red 0.07563035
green 0
blue 0

4. Drag the **red** slider to adjust the color values. When you are finished, press **R** over the Viewer to display all channels.

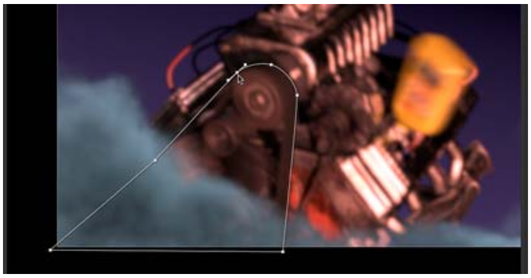
   The Exposure node worked as expected, but the result is less than spectacular. The color change is too uniform. If only there were a way to limit—or, in fact, *mask*—the color correction, perhaps we'd see a better composite. Hmm...

Masking Effects

You can, indeed, apply masks to limit how each of these nodes affects the images. The following shows how to create a Bezier mask to limit the color-correction.

To Create and Apply a Bezier Mask

1. Click on a blank space in the node graph, so that nothing is selected in the node tree.
2. From the Toolbar, choose **Draw > Roto** to insert a Roto node.
3. Click inside the Viewer window to draw a Bezier shape over the image, like this:



4. To refine the shape, click on a point to select it and then drag to adjust its position.
5. To create sharp corners, select a point, right-click and choose **Cusp** from the pop-up menu.
6. To add points to the shape, simply select the **Add Points** tool and click on the shape's outline.
7. When you're satisfied with the shape, drag the **mask** connector from the **EXPTool1** node to the output of the **Roto** node.

   In the EXPTool1 control panel, the mask channel option is now set to the **rgba.alpha** channel of the node that is connected to the **mask** input. In this case, this is the alpha channel of the Roto node.

Creating Flipbook Previews

On the Viewer window, the timeline buttons let you play the project, but if you pay attention to the frames-per-second (FPS) field at the top of the Viewer window, you may notice that Nuke doesn't provide real-time playback. This is because Nuke renders on-the-fly to display images in the Viewer. It's fast, but also limited by the amount of memory and computer-processing power available to you.

The Flipbook feature provides better real time preview, because it is prerendered for the FrameCycler viewer, included with Nuke. Keep in mind that the Flipbook feature will render a preview that matches the active resolution; if you're in proxy mode, for example, that's the resolution you'll get in the flipbook.

Note *The Flipbook feature renders temporary files in the directory you specified for "disk cache" under* **Nuke > Preferences**. *You'll also find an option there* *that allows you to limit the amount of disk space that the flipbook feature may use.*

To generate a flipbook:

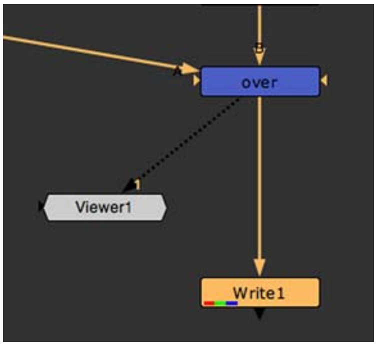
1. Select the **Over** node at the bottom of your node tree.
2. From the menu bar, choose **Render > Flipbook selected**.
3. Enter **1-28** as the number of frames to preview and click **OK**.
4. When the flipbook is ready to view, a copy of the FrameCycler window will appear. Click the **Play** button to view the results.
5. Close the FrameCycler window to return to your project.

Rendering Final Output

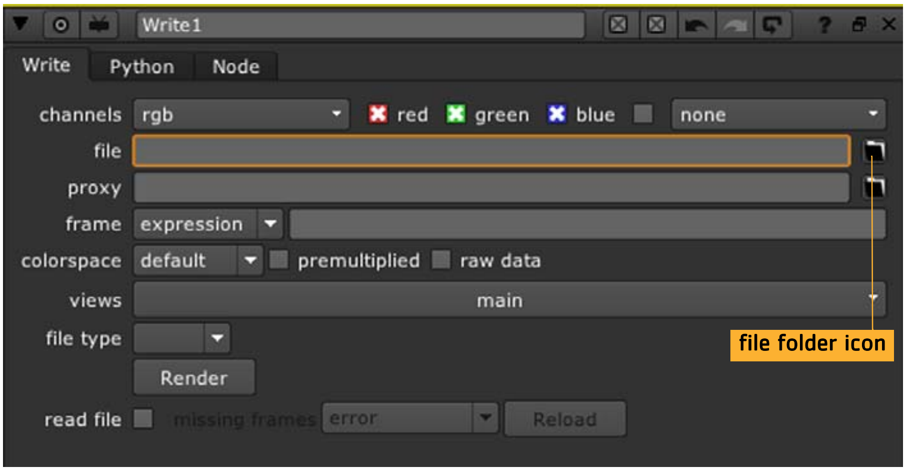
When you're ready to render the results of your composite, you insert a Write at the bottom of the node tree, and specify the pathname for the rendered images. Although we'll use just one here, you can place several Write nodes in your script, anywhere you like, to render output from different places in the tree. When the render order is important, use the **render order** option in the Write nodes to specify the order in which multiple renders should be executed.

To render the result of your composite:

1. Select the last **Over** node at the bottom of the node tree.
2. Right-click and choose **Image > Write** to add a node for output.

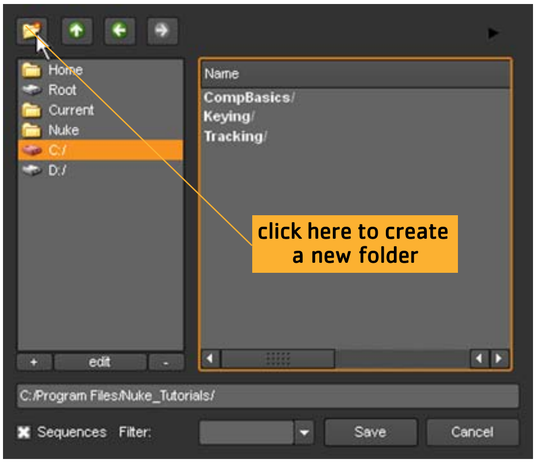


3. In the control panel for the Write node, click the **file** folder icon.


Write1
Write
Python
Node
channels
rgb
☒ red
☒ green
☒ blue
☐
none
file
proxy
frame
expression
colorspace
default
☐ premultiplied
☐ raw data
views
main
file type
Render
read file ☐
missing frames
error
Reload

file folder icon

4. Browse to the **Nuke_Tutorials** directory.


Home
Root
Current
Nuke
C:/
D:/
Name
CompBasics/
Keying/
Tracking/
+
edit
-
☒ Sequences
Filter:
Save
Cancel

C:/Program Files/Nuke_Tutorials/

click here to create a new folder

5. Click the "new folder" icon, in the upper-left corner of the browser, and type **Rendered** as the name for the new folder. Click **OK**.

6. Select the folder you just created.

   You should see the "Nuke_Tutorials/Rendered/" pathname displayed at the bottom of the browser.

7. At the end of the "Nuke_Tutorials/Rendered" pathname, type **first_comp.####.exr** as the name for the rendered image sequence, and then click **Save**.

8. Choose **Render > Render All** to render the images, or simply click the **Render** button inside the Write control panel.

9. Nuke prompts you to specify the frames to render. Enter **1-28** as the frame range and click **OK**.

A status window appears that shows the progress of your render. When the render is complete, you'll find the sequential images in the "Nuke_Tutorials/ Rendered" directory. To check the results, simply insert a new Read node, point to the new image sequence, and then generate a flipbook with the Read node selected.

Using the Nuke frame number variable

What's that "####" bit in the filename, you say? That's the variable that tells Nuke where to place the sequential numbers or frame numbers. You only type one name to represent the image sequence, but Nuke will create one image file for each frame in your shot.

So, in this case, you entered "first_comp.####.exr" but Nuke will render these files for frames 1 through 5: "first_comp.0001.exr," "first_comp.0002.exr," "first_comp.0003.exr," "first_comp.0004.exr," and "first_comp.0005.exr." You can change the number of hash marks in the variable—##, ###, #####—to change the number of padded digits for the frame numbers.

Image formats

If you don't specify a file format inside the Write node control panel, Nuke uses the format specified by the filename extension you typed. For example, in this tutorial, you used the ".exr" extension to tell Nuke to save the images as OpenEXR files.

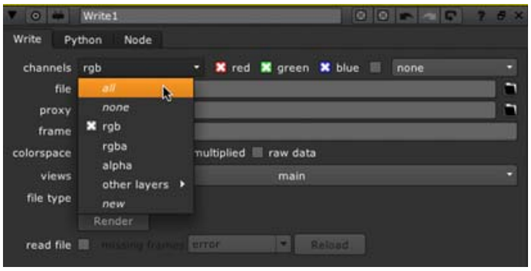
Rendering with the active resolution

When you execute a render or a flipbook, Nuke assumes you want to render the active resolution. When you're in full-res mode, for example, Nuke renders full-resolution images to disk. When you're in proxy mode, Nuke assumes you want to render the proxy resolution—defined in the Project Settings window—to the path and filename you specified as the **proxy** filename in the Write node. If the proxy field is empty or pointing to an invalid path, Nuke returns an error.

It's easy to toggle to proxy mode and then forget your images will be rendered in the lower resolution. Before you execute a render, it's always a good idea to check which resolution is active. In the Viewer, the label at the lower-right corner of your image will indicate whether you are in full-res or proxy mode.

Rendering multiple channels

When you insert a Write node, Nuke assumes that you need only the RGB channels in the final render. In many cases, this is acceptable because you won't need the alpha channel or other channels from the node tree when you deliver final shots to your clients. However, sometimes you need to render intermediate files—such as mattes, projection elements, or subcomps—and include all the channels in your node tree.


Write1
Write
Python
Node
channels
rgb
☒ red
☒ green
☒ blue
☐
none
file
proxy
frame
colorspace
views
file type
all
none
☒ rgb
rgba
alpha
other layers
new
Render
multiplied
☐ raw data
main
read file ☐
missing frames
error
Reload

For example, rather than manage several elements for an animated character, you could combine the character animation, the lighting passes, alpha channel, and a depth mask in one image sequence on disk. This makes it easier to manage elements in the final composite and simplifies the artist's workflow.

To output all channels, change the Write node's Output list from **RGB** to **All Channels**, select the OpenEXR file format, and then execute the render. Currently the OpenEXR format (.exr) is the only file format that supports unlimited channels.

Epilog

In this tutorial, you setup a new project and created a simple composite. You learned how to use (or at least, locate) practically every Nuke window and tool, and you rendered out the result of your composite. You're finished! Go home!

Well... there might be a few more things you want to know. After this tutorial, you should feel comfortable with the Nuke user interface, so put on your explorer hat and review the other tutorials. There's no specific order from here, so look through the following pages until you find what interests you.

- End of Tutorial -

TUTORIAL 2: TRACKING, STABILISING, AND MATCHMOVING

Every filmmaker knows the challenges of putting together a vision. You may not have the money to build post-apocalyptic Montreal, but you might have enough to create it in post. You may have brilliant performances by your actors—but not together in the same shot. Fortunately, you can composite the best takes. Your battle sequence with 5 A-list actors, 100,000 extras and 57 elephants, comes back from the lab with scratches on the negative. You can fix it. You can. A savvy production team knows how to leverage digital technology to make it possible, and Nuke's tracking tools are indispensable for these situations.

As you may know, tracking is the process of recording the location of features as they move through the scene. The result is stored as 2D coordinates on the image plane. Once you have the tracking data, you can use the movement to perform a variety of useful tasks, such as stabilizing the footage, applying the movement to other elements in your composite, and improving the accuracy of roto mattes.

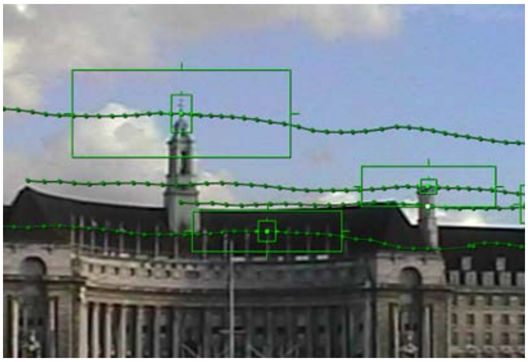


Figure 2.1: Tracking image features.

An important aspect of the tracking process involves carefully reviewing your footage *before* you attempt to track. Play your images several times—preferably with a flipbook—and look at the direction of movement for the features you want to track. Note potential problems with motion blur, obscuring objects, or frames where the features are hidden or move off screen.

Tip *Nuke can often compensate for "problem footage," but tracking works best when you can identify distinct features throughout the length of the shot.*

One-Point, Two-Point, Three-Point, Four

Before we get into the first example, let's review a few tracking concepts. You can track up to four distinct features or patterns with each Tracker node in Nuke. How do you decide whether to track one, two, or more features? It depends on what you want to do with the data and the level of accuracy you need in the result. Here are some general guidelines:

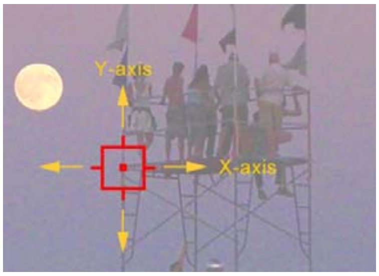


Figure 2.2: One track: X and Y position only.

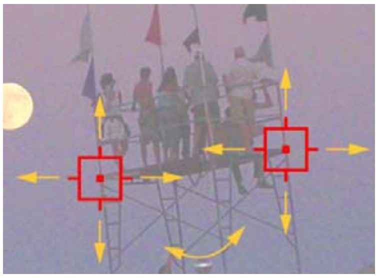


Figure 2.3: Two tracks: X, Y, and Z-rotation.

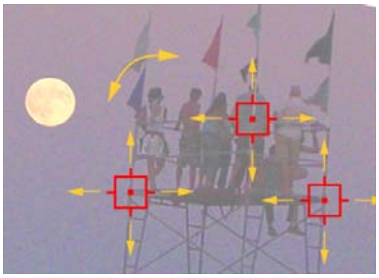


Figure 2.4: Three tracks: X, Y, Z-rotation, & scale.

- **One-point tracking**. Track one feature's horizontal (x-axis) and vertical (y-axis) position, with little or no perspective change on the image. You can apply this information to move other elements in the composite or apply the inverse to stabilize the image.
- **Two-point tracking**. Track horizontal and vertical position for two features. The feature positions, relative to each other, indicate whether the image is rotating clockwise or counter-clockwise (z-axis rotation). In some cases, two tracking points is sufficient to calculate the scaling of the features, as well.
- **Three-point tracking**. Track horizontal and vertical position for three features. Provides all the benefits of two-point tracking with an additional set of tracking data for more accuracy on z-rotation and scaling.
- **Four-point tracking**. Again, all the benefits of the lesser tracks with an additional set of tracking data. Three-point is usually sufficient for most 2D tracking needs, but four-point makes it possible to distort and

matchmove another element into the four points, or corners, of the features you track. That's why four-point tracking is typically called *cornerpin* tracking.

Open the Tutorial Project File

In this tutorial, you work from a project file that already includes the node trees. Each tree is setup for the examples that follow.

To open the project file:

1. Launch the Nuke application and choose **File > Open** from the menu bar.
2. In the file browser, navigate to your **Nuke_Tutorials/Tracking/** folder, select the **tracking_tutor.nk** project file and click **Open**.
3. It will show some nodes in error. Don't worry! It can't find the tutorial files. So before doing anything else you have to tell this script where to find these tutorial images, assuming you can remember where you put them. Double-click on the NoOp node in the top left corner of your node graph. It's called Tutorial_Path. Double clicking will bring up a property panel on the right. Enter the path to the tutorial files in the Tutorial Project Directory. Use the file browser as that's often easier than typing it in. You should then see tutorial images appear.
4. Move the mouse pointer over the node graph, and press **F** to frame the entire contents of the project file.

The examples in this project file are grouped with colored boxes, called *backdrops*, and each contains a node tree for the tutorial examples that follow.

Tip *Backdrops let you organize groups of nodes, like those shown in this project file. Choose* **Other > Backdrop** *from the Toolbar. Drag the backdrop title bar to move it. Drag the backdrop corner to resize it. Any nodes surrounded by the borders of the backdrop will move with the backdrop when you drag its title bar.*

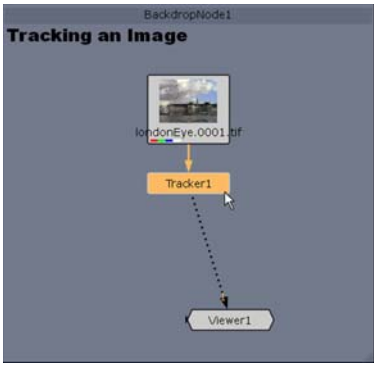
Tracking a Single Feature

In this first example you'll learn how to track a single feature, which is the most basic 2D tracking operation. After you achieve a solid track for one feature, you can build on that and track other features as needed.

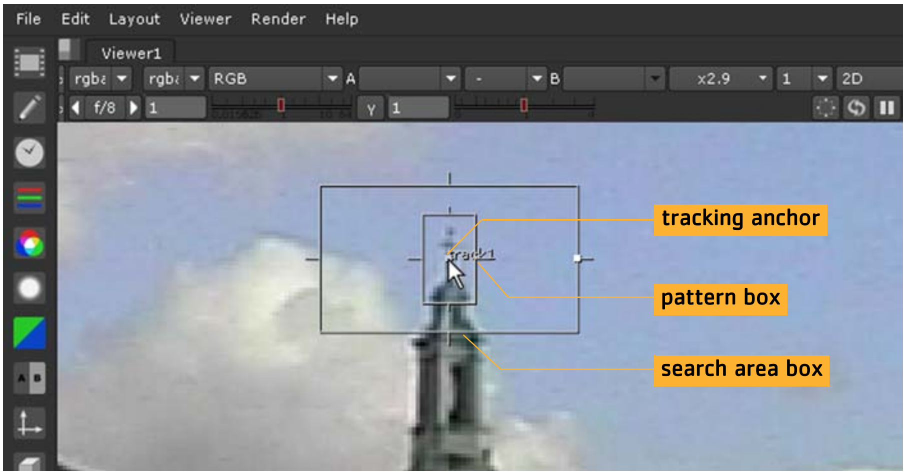
To track one feature (one-point tracking):

1. In the project workspace for the **tracking_tutor.nk** file, locate the node tree labelled "**Tracking an Image**."

2. Click on the **LondonEye** Read node to select it.
3. From the menu bar, choose **Render > Flipbook selected**. When the FrameCycler window appears, play the flipbook several times to review the footage.
4. Look at the features in the image and notice the amount and direction of movement as the clip plays. When you're done, minimize the FrameCycler window.
5. Choose **Transform > Tracker** from the Toolbar to attach a new Tracker node to the **LondonEye** Read node.

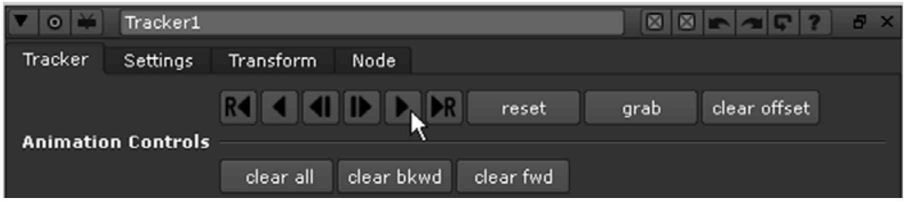

BackdropNode1
Tracking an Image
londonEye.0001.tif
Tracker1
Viewer1

6. Connect the **Viewer1** node to the **Tracker1** node. Inside the Viewer, you'll see one tracking marker.
7. In the Viewer, scrub the time slider to frame 1, to make sure you're at the beginning of the shot.
8. Use the mouse to drag the tracking anchor over the tower spire, shown below:

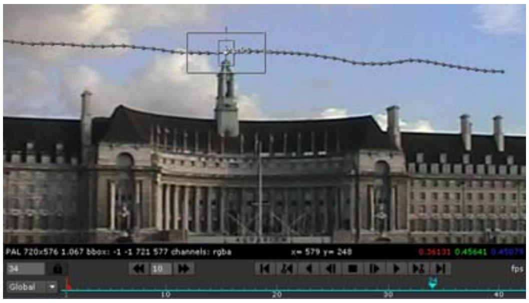

File
Edit
Layout
Viewer
Render
Help
Viewer1
rgba
rgba
RGB
A
-
B
x2.9
1
2D
f/8
1
Y
1
track1
tracking anchor
pattern box
search area box

9. Click on the pattern box (inner box) of the tracking marker, and adjust its size to contain the feature.
10. Click the search area (outer box) of the tracking marker, and adjust its size to enclose the amount of space you think the feature may move between frames.

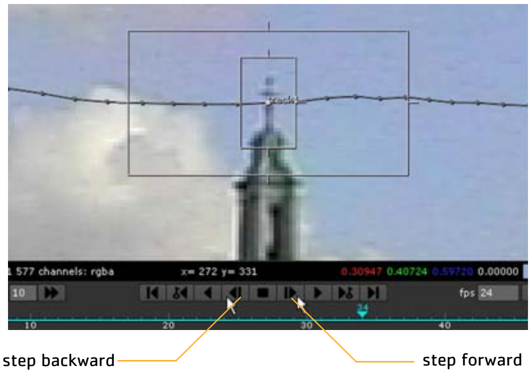
    Large search areas require more calculation time, so keep it as small as possible. However, when the search area is *too* small, the feature may move outside the box and you'll lose the track. If you aren't sure how large to make the search area, go back and review the flipbook of your image.
11. In the control panel, click the "track forward" button to generate the track.


Tracker1
Tracker
Settings
Transform
Node
reset
grab
clear offset
Animation Controls
clear all
clear bkwd
clear fwd

When completed, you'll see the track curve with points that mark the keyframe positions along the curve.



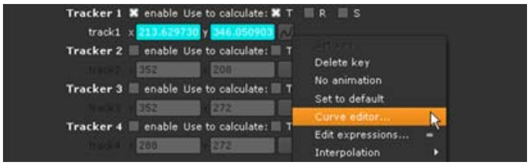
12. Use the "next frame" and "previous frame" buttons on the timeline to step through the timeline and verify the accuracy of the track.


track1
577 channels: rgba
x= 272 y= 331
0.30947 0.40724 0.59720 0.00000
10
fps 24
10
20
30
40
step backward
step forward

The track is fairly solid in this example. However, some images don't track as easily as this one. When you need to edit track data, what do you do? You open the Curve Editor of course! Let's assume you need to smooth the points of this track.

To edit track data:

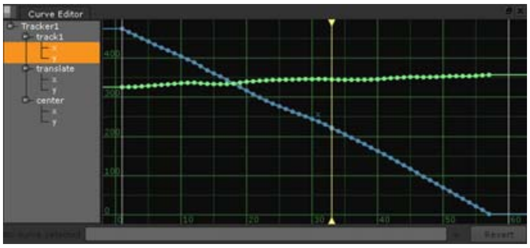
1. In the tracker control panel, click the animation button next to the **Tracker 1** parameter, and choose **Curve editor** from the pop-up menu.



Tracker 1 ☒ enable Use to calculate: ☒ T ☐ R ☐ S
track1 x 213.629730 y 346.050903
Tracker 2 ☐ enable Use to calculate: ☐ T
track2 x 352 y 208
Tracker 3 ☐ enable Use to calculate: ☐ T
track3 x 352 y 272
Tracker 4 ☐ enable Use to calculate: ☐ T
track4 x 288 y 272

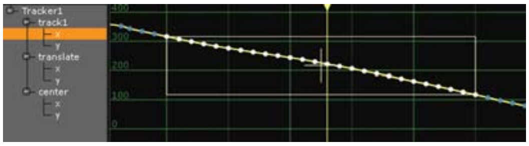
Delete key
No animation
Set to default
Curve editor...
Edit expressions...
Interpolation

2. Click the items in the Curve Editor outline and you'll see values recorded for each of the parameters during the tracking process.

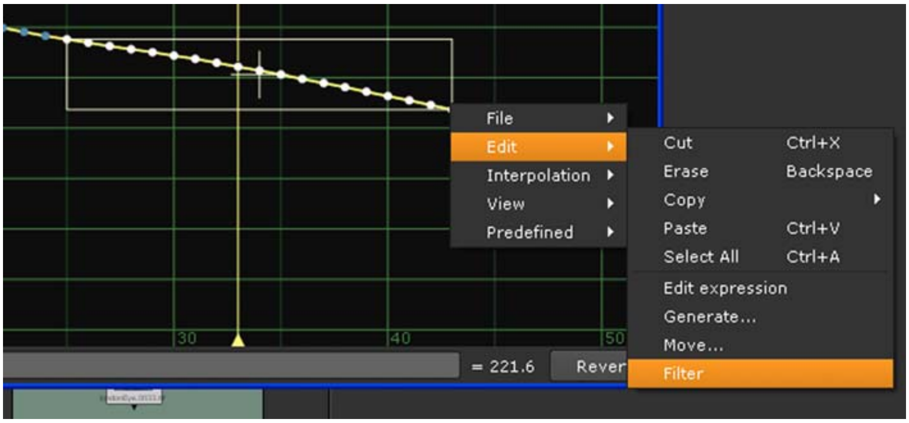

Curve Editor
Tracker1
track1
x
y
translate
x
y
center
x
y
400
300
200
100
10
20
30
40
50
Revert

3. Hold down the **Shift** key and click the **X** and **Y** curves, under **track1**, to select both curves at once and press the **F** key to "frame" the curves.
   - To adjust a value, select a point and drag it up or down.
   - To change the frame for a particular point, select it, hold down the Ctrl key and drag the point left or right.

4. Let's assume you want to smooth a curve by applying a filter to the values. Draw a marquee—drag while pressing the left mouse button—around a section of the curve to select multiple points.


Tracker1
track1
x
y
translate
x
y
center
x
y
400
300
200
100
0

5. Click the right mouse button to display a pop-up menu of Curve Editor options. Choose **Edit > Filter** and enter **2** as the number of times to filter the key frames.


File
Edit
Interpolation
View
Predefined
Cut Ctrl+X
Erase Backspace
Copy
Paste Ctrl+V
Select All Ctrl+A
Edit expression
Generate...
Move...
Filter
30
40
50
= 221.6
Rever

Nuke averages the location of each point based on the values of the surrounding points, and this helps to smooth the curve.

6. Close the Curve Editor window and then play the result in the Viewer.

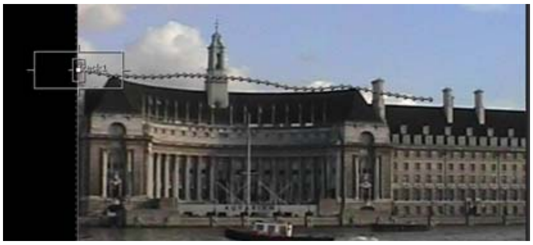
   Those are the basics for tracking and editing the results for a single feature. In the next example, we'll make it a little harder—tracking a feature that moves out of view.

Tracking Obscured Features

At the end of the previous example, you may have noticed the track was dropped at frame 58 when the feature moved off the screen. Disappointing? Yes. A disaster? Probably not. When features move out of frame, or become obscured by other elements in the image, you can use the track offset feature to pass the tracking operation to another feature in the image. Nuke then attempts to continue the track along its current course.

To track a feature that moves off screen:

1. In the project workspace, locate the node tree "**Tracking Obscured Features**."
2. Double-click the **Tracker2** node to open its control panel. This node tracks one of the chimneys in the clip you used from the previous example.
3. Attach a Viewer to the **Tracker2** node and scrub the timeline until you see the tracked feature move out of frame.

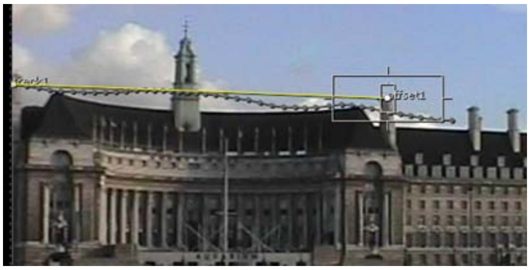


   As you can see, track1 accurately tracks its feature through most of the clip—until it moves off the screen at frame 44. This is where the problem starts.

4. Press the plus key (**+**) on your keyboard a few times to zoom in on the Viewer.

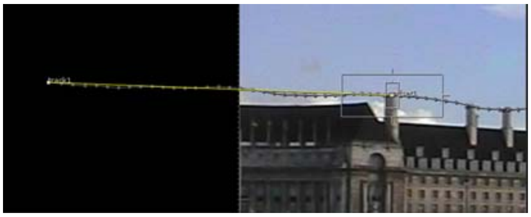
   You want to select an alternate feature that stays in view during the length of the clip.

5. At frame 44, press the **Ctrl** (or **Command**) key and drag the **track1** point to the first chimney at the right.



   The "offset1" label appears with a line connecting the new feature to the original feature.

6. In the **Tracker2** control panel, press the "track forward" button and Nuke continues the track off the screen.

   Why is this cool? Because you can now use the track data to matchmove an element—a trail of chimney smoke, for example—that will lock to the feature even after it moves off the screen.
7. The track is now complete, so you can click the **clear offset** button in the Tracker2 control panel.
8. Uncheck the **enable** box for **track1** to prevent it from being recalculated.
9. Before you continue, close all Tracker control panels that are currently open.

The offset doesn't change the track location. Instead, it allows Nuke to continue the track with the assumption that the offset feature remains at the same relative distance to the original feature. Later in this chapter, you'll see how to use this tracking data to composite another element to match the background plate.

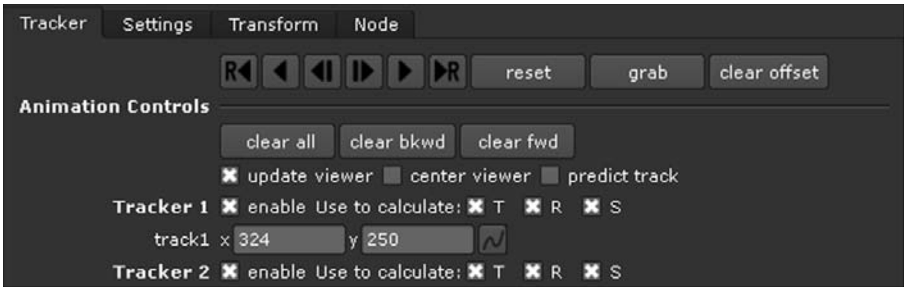
Stabilising Elements

Stabilisation is the process of removing motion—camera-shake, for example—and locking down the element for your composite. A one-point track provides enough information to stabilize horizontal and vertical motion along the image plane. A two-point track lets you stabilize horizontal and vertical motion, and remove rotation in the image, as well.

To track and stabilize:

1. Locate the node tree labelled "**Stabilizing Elements**."
2. You'll see a copy of the same **LondonEye** Read node that we've been using for the other examples. Click on it to select it.
3. Choose **Transform > Tracker** and then attach a Viewer to the new **Tracker3** node.
4. In the control panel for Tracker3, check the boxes to **enable Tracker1** and **Tracker2**.

5. For each track, check the boxes for **T** (translate), **R** (rotate) and **S** (scale).


Tracker
Settings
Transform
Node
R◀
◀
◀|
|▶
▶
▶R
reset
grab
clear offset
Animation Controls
clear all
clear bkwd
clear fwd

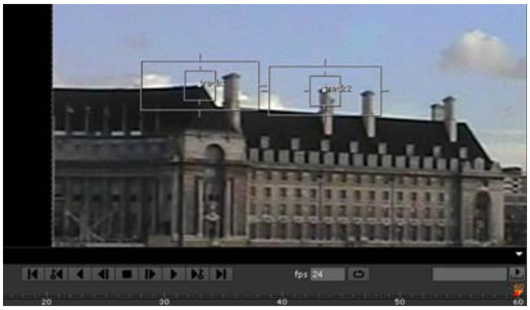
☒ update viewer ☐ center viewer ☐ predict track

Tracker 1 ☒ enable Use to calculate: ☒ T ☒ R ☒ S

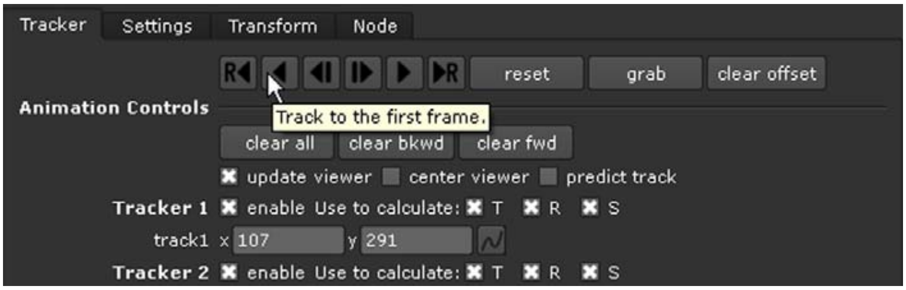
track1 x 324 y 250

Tracker 2 ☒ enable Use to calculate: ☒ T ☒ R ☒ S

6. In the Viewer, scrub to the end of the timeline and adjust the size and position of each tracking marker for the features shown below:


track1
track2
fps 24
20
30
40
50
60

7. In the control panel, click the "track backward" button to generate the tracks.


Tracker
Settings
Transform
Node
R◀
◀
◀|
|▶
▶
▶R
reset
grab
clear offset
Animation Controls
Track to the first frame.
clear all
clear bkwd
clear fwd

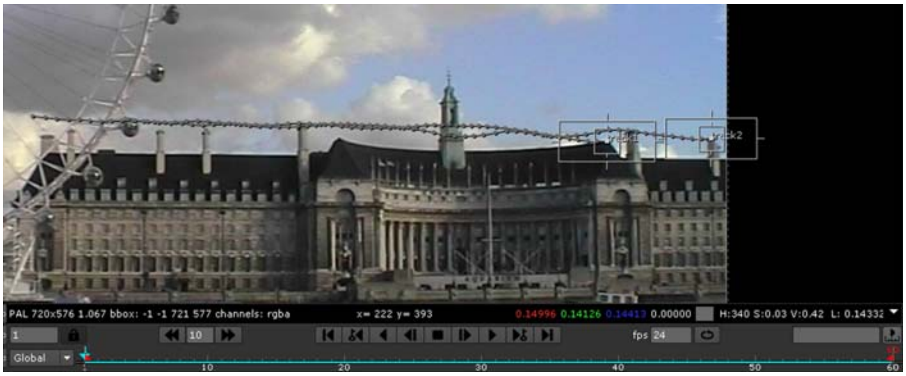
☒ update viewer ☐ center viewer ☐ predict track

Tracker 1 ☒ enable Use to calculate: ☒ T ☒ R ☒ S

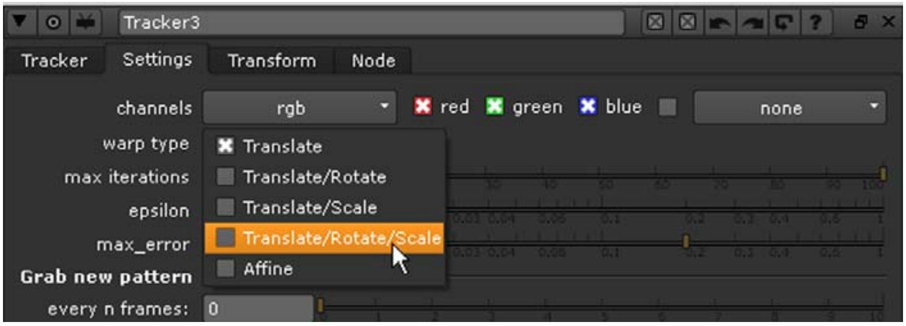
track1 x 107 y 291

Tracker 2 ☒ enable Use to calculate: ☒ T ☒ R ☒ S

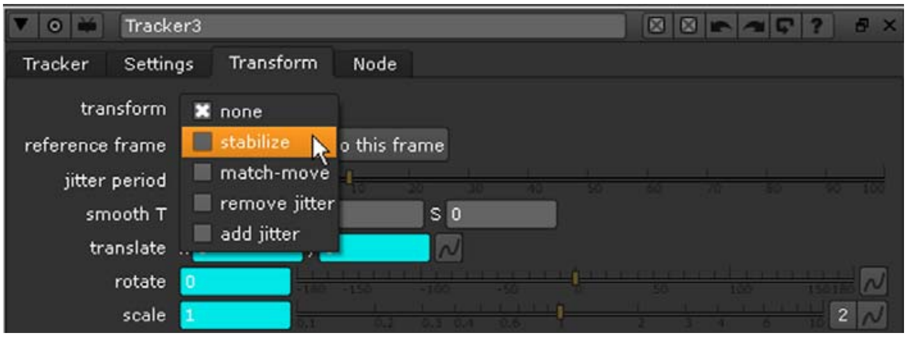
Now you have the position data for two tracks, and you can use this information to remove the unwanted movement in the image.


PAL 720x576 1.067 bbox: -1 -1 721 577 channels: rgba
x= 222 y= 393
0.14996 0.14126 0.14413 0.00000
H:340 S:0.03 V:0.42 L: 0.14332
Global
fps 24

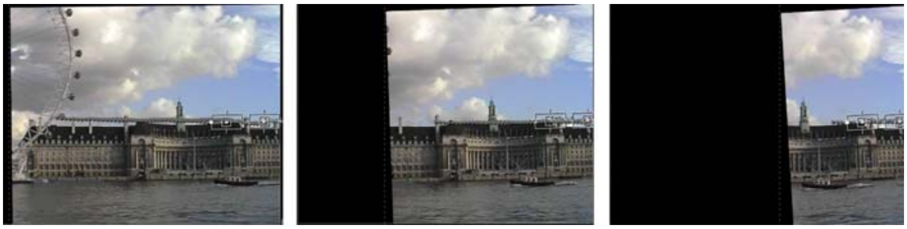
8. In the Tracker3 control panel, click the **Settings** tab and choose **Translate/Rotate/Scale** from the **warp type** list.


Tracker3
Tracker
Settings
Transform
Node
channels
rgb
red
green
blue
none
warp type
Translate
Translate/Rotate
Translate/Scale
Translate/Rotate/Scale
Affine
max iterations
epsilon
max_error
Grab new pattern
every n frames:
0

9. Click the **Transform** tab and choose **stabilize** from the **transform** list.


Tracker3
Tracker
Settings
Transform
Node
transform
none
stabilize
match-move
remove jitter
add jitter
reference frame
o this frame
jitter period
smooth T
S 0
translate
rotate
0
scale
1

10. In the Viewer, click the "play forward" button and review the results.



As the clip plays, you'll see the features remain locked to the same position within the compositing frame.

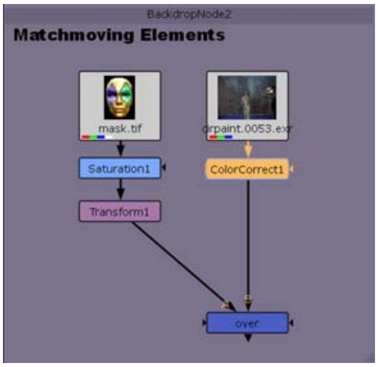
Tip *After you track and stabilize footage, you can add a **Transform** > **Transform** node after the Tracker3 node to adjust the position and the rotation of the stabilized image for a final composite.*

Matchmoving Elements

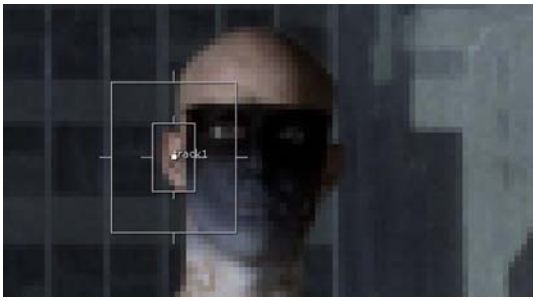
Matchmoving is the opposite of stabilisation. The intent is to record and use the motion in an image and apply it to another element. In the following example, you'll use the tracker to matchmove and composite a mask image onto the performer in a background plate.

To matchmove an element:

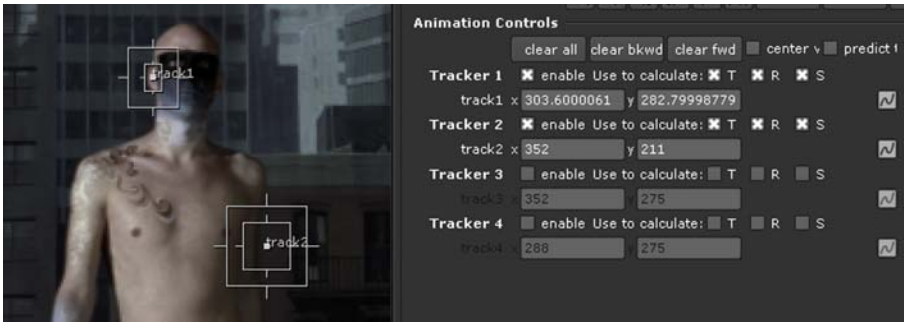
1. Find the node tree labelled "**Matchmoving Elements**."
2. Drag the time slider to the beginning of the timeline. Select the **ColorCorrect1** node and then choose **Transform > Tracker**.



3. Attach a Viewer to the new **Tracker4** node and position the **track1** marker over the performer's right ear.



4. Adjust the size of the pattern box and the search area as shown.
5. In the control panel, next to Tracker1, check the boxes for **T** (translate), **R** (rotate) and **S** (scale).
6. Click the **enable** box for **Track 2** to activate the controls for an additional track. Check the boxes for **T** (translate), **R** (rotate) and **S** (scale) on this track, also.
7. Press the minus key (**-**) over the Viewer to zoom out, and you'll see a second tracking marker appears.



Animation Controls

☐ center v ☐ predict

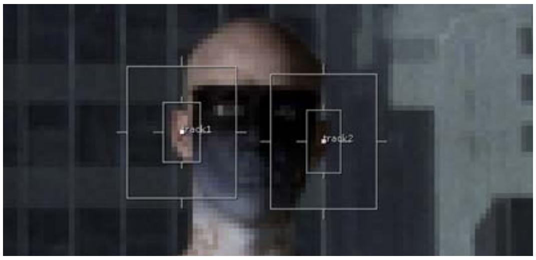
Tracker 1 ☒ enable Use to calculate: ☒ T ☒ R ☒ S
track1 x 303.6000061 y 282.79998779

Tracker 2 ☒ enable Use to calculate: ☒ T ☒ R ☒ S
track2 x 352 y 211

Tracker 3 ☐ enable Use to calculate: ☐ T ☐ R ☐ S
track3 x 352 y 275

Tracker 4 ☐ enable Use to calculate: ☐ T ☐ R ☐ S
track4 x 288 y 275

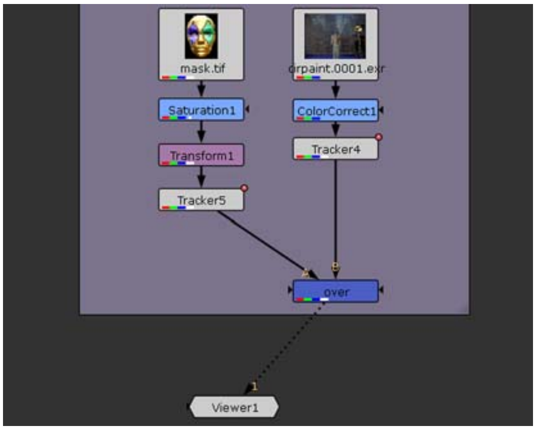
8. Adjust the size and position of the second tracking marker, as shown below.



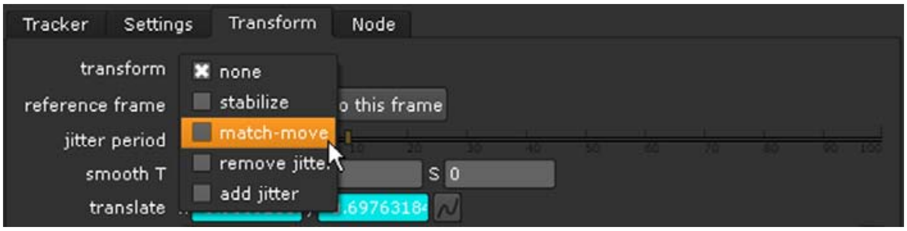
9. In the control panel for Tracker4, click the "track forward" button to generate the tracks.

   After you get a solid track on the performer, you need to make a copy of the Tracker4 node to create the matchmove.

10. Select the **Tracker4** node and press **Ctrl+C** to copy it.
11. Select the **Transform1** node and press **Ctrl+V** to paste the Tracker node copy (Tracker5).
12. Connect the Viewer to the **Over** node. Your node tree should now look similar to this:


mask.tif
orpaint.0001.exr
Saturation1
ColorCorrect1
Transform1
Tracker4
Tracker5
over
Viewer1

13. In the Tracker5 control panel, click the **Transform** tab and choose **match-move**. Then close the Tracker5 control panel.


Tracker
Settings
Transform
Node
transform
reference frame
jitter period
smooth T
translate
none
stabilize
match-move
remove jitte
add jitter
o this frame
S 0
.6976318

14. Click the "play forward" button in the Viewer or render a flipbook and you should see the Mardi Gras mask transform to match the movement of the performer.

If you see jitter in the movement, you can edit the track data in the Curve Editor to smooth out the data. You can also add a small value to the de-jitter parameter or add values to the smooth T, R, and S parameters on the Transform tab to filter the tracks.

Epilog

In this tutorial, you worked with several examples for the Tracker node. You learned how to record the locations for multiple features and you applied the tracking data for other tasks in the composite, such as stabilisation and matchmoving.

- End of Tutorial -

TUTORIAL 3: KEYING AND MATTES

Keying is one of those fundamental compositing skills. You can't composite anything until you have mattes pulled for the elements you want to layer together. It's nice to say you could just push a button to complete this task, but as you probably know, one keying operation seldom produces an acceptable matte. Image quality, lighting conditions, subject motion, colors—even camera moves—affect the steps required to get a clean matte for your composite.

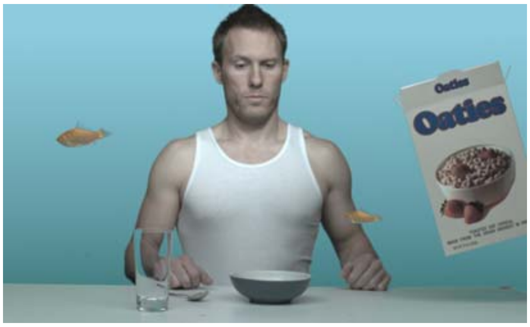


Figure 3.1: Keying Footage in Nuke.

So how do you get a clean matte in Nuke? The best approach is to understand the strengths of each keying tool and combine them as needed. This tutorial shows how to pull keys in Nuke and how to layer the results with channel operations, merge nodes, and rotoshapes.

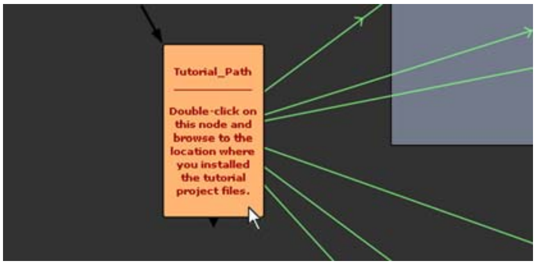
Open the Tutorial Project File

The project file for this tutorial includes several node trees for the keying operations described in this chapter.

To open the project file:

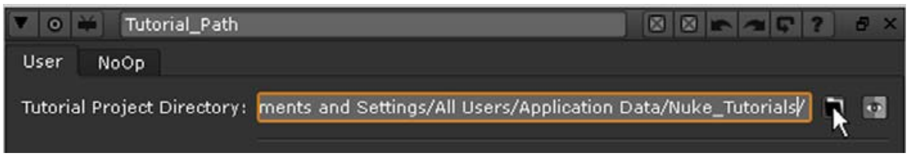
1. Launch the Nuke application and choose **File > Open** from the menu bar.
2. In the file browser, navigate to your **Nuke_Tutorials/Keying/** folder, select the **keying_tutor.nk** project file and click **Open**.

3. Double-click on the **Tutorial_Path** node, located on the left side of the script, to open its control panel.



Double-click on this node and browse to the location where you installed the tutorial project files.

4. In the **Tutorial_Path** control panel, click the "file folder" button. Browse to the location where you installed the tutorial project files, and then click **Open** to select the location.



Tutorial Project Directory: ments and Settings/All Users/Application Data/Nuke_Tutorials/

   After you select the correct path, the error messages should clear from the Read nodes, and the thumbnails in the script will update with the correct images.

5. Close the **Tutorial_Path** control panel. Then, choose **File > Save As** to save a copy of the project file.
6. Move the mouse pointer over the node graph, and press **F** to frame the entire contents of the project file.

   The green arrows (lines) show the links between the Tutorial_Path node and the Read nodes.

7. If you wish, press **Alt+E** to hide the expression arrows.

   The Tutorial_Path node saves the location of the project files on your computer, so you don't need to repeat this for future sessions.

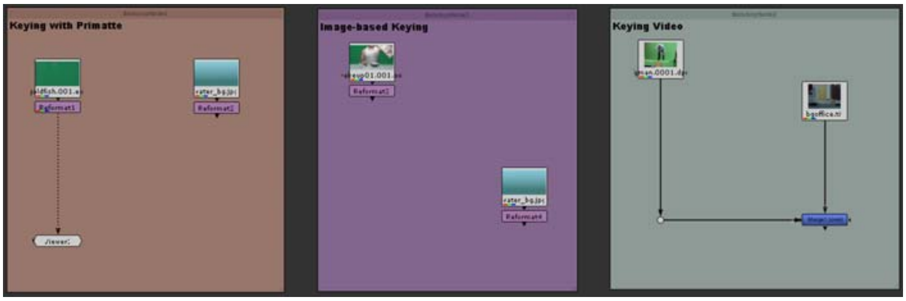


Figure 3.2: Node trees in the keying_tutor.nk project file.

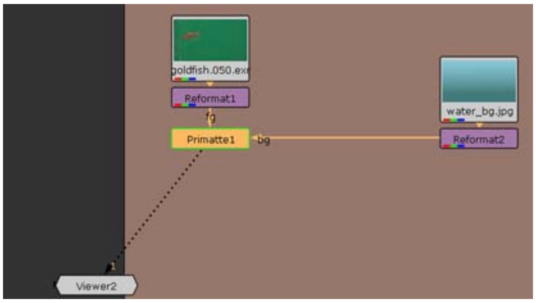
Keying with Primatte

The Primatte keyer includes a quick "Auto-Compute" option that evaluates your image and determines a good baseline key. From there, you can easily tweak the settings and generate an acceptable matte.

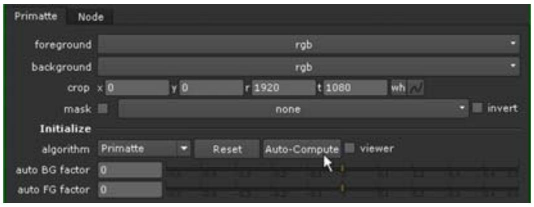
The two examples in this section show how to pull a key with the Auto-Compute option (method 1), and also how to manually sample a color from the screen background and build your key from there (method 2).

To pull a key with Primatte (method 1):

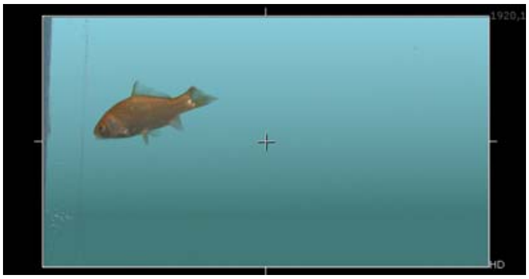
1. In the project file, locate the node tree labeled "**Keying with Primatte**," and make sure a Viewer is attached to the **Reformat1** node.
2. Choose **Keyer > Primatte** to insert the keyer between the foreground image and the Viewer.



3. Drag the **bg** connector from **Primatte1** to the **Reformat2** node, which supplies the background image for this example. The **fg** connector should be attached to **Reformat1**.
4. Move the time slider to frame **50**, and click the **Auto-Compute** button inside the **Primatte1** control panel.



foreground rgb
background rgb
crop x 0 y 0 r 1920 t 1080 wh
mask ☐ none ☐ invert
algorithm Primatte
☐ viewer
auto BG factor 0
auto FG factor 0



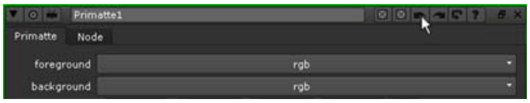
That's it. You're done... well, nearly done. We need a "free-floating" goldfish, but the reflections in the aquarium glass clearly indicate "captivity."

A garbage matte will easily remove the reflections, and you'll learn how to do that later in the section on rotoscoping. For now, let's keep working with Primatte.

As you've seen, Primatte's auto-compute option can quickly pull keys on certain images. However, you should also know how to pull and tweak keys manually. You might, for example, need more control over the transparency of the fins on the goldfish.

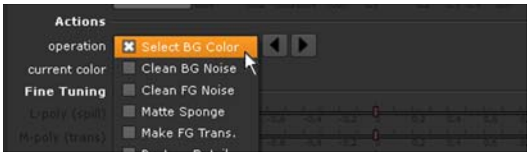
To pull a key with Primatte (method 2):

1. Continuing from the previous example, open the **Primatte1** control panel.
2. Click the "undo" button at the top of the control panel to step back to the previous state of the **Primatte1** node. Or, you can also delete the current **Primatte1** node and insert a new one.

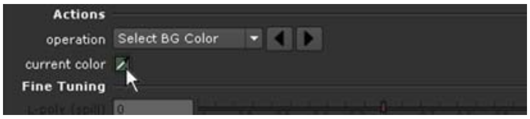

Primatte1
Primatte
Node

foreground rgb
background rgb

3. Scroll down through the Primatte options and set the keying **operation** to **Select BG Color**.

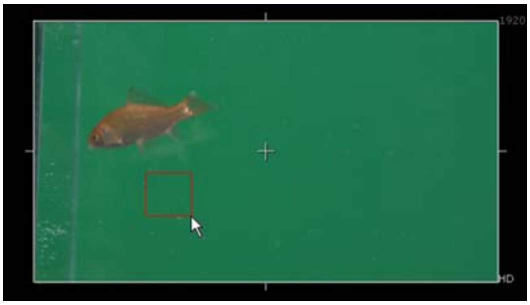

Actions
operation
current color
Fine Tuning
Select BG Color
Clean BG Noise
Clean FG Noise
Matte Sponge
Make FG Trans.

4. The **current color** chip should display the eyedropper icon. If it doesn't, click on the color chip to toggle the eyedropper.


Actions
current color
Fine Tuning

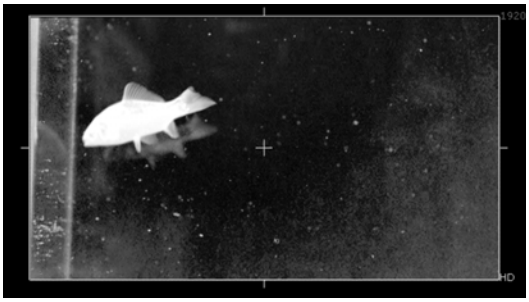
operation Select BG Color

5. Hold down the **Ctrl+Shift** keys (Mac users, hold down **Command+Shift**) and drag—or scrub—over a portion of the greenscreen in the image displayed in the Viewer.



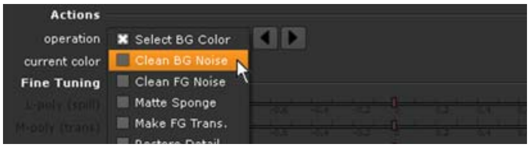
This returns an average color-pick of the sampled pixels. If you want a color pick from a single pixel, press **Ctrl** or **Command** and click once over the greenscreen. After you pick, you can clear the red square by **Ctrl-** or **Command-**clicking again.

6. Press **A** over the Viewer to toggle to the alpha channel display. Looks like the aquarium is not as clean as we thought. Our color pick gave us a fairly noisy key, so let's clean it up.

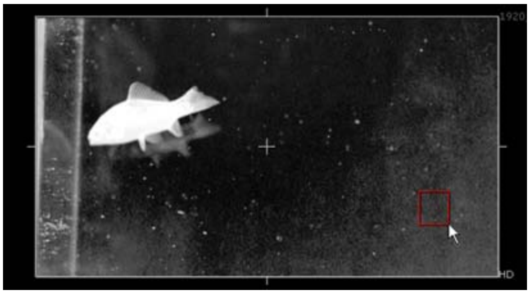


Now you'll sample a few areas of the image to "push" selected pixels to one of three areas: the transparent matte, the opaque subject, or the semi-transparent part of the matte.

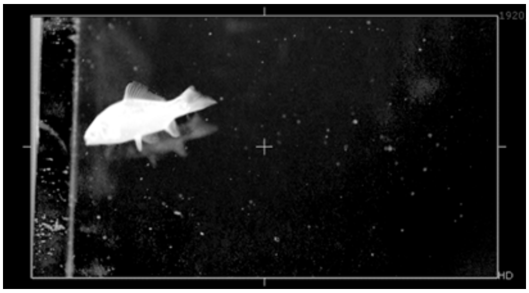
7. In the **Primatte1** control panel, change the keying **operation** to **Clean BG Noise**.


Actions
operation
current color
Fine Tuning
Select BG Color
Clean BG Noise
Clean FG Noise
Matte Sponge
Make FG Trans.
Restore Detail

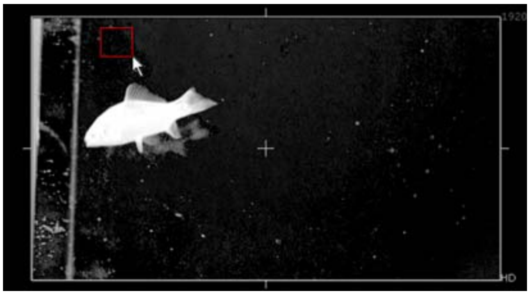
8. Press **Ctrl+Shift** or **Command+Shift** and drag a small square over the dark area in the lower-right corner of the image.



This second color sample cleans the background by “pushing” the selected pixels into the transparent area of the matte. You probably need a few more samples to get a better key.



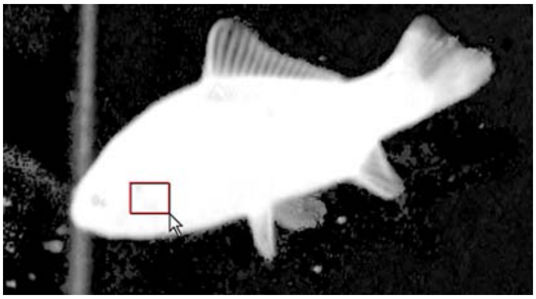
9. Scrub a few small areas in the background, focusing on the gray pixels until the matte improves.



The background doesn’t need to be solid black. We’re just trying to get a good separation between our foreground subject and the greenscreen background.

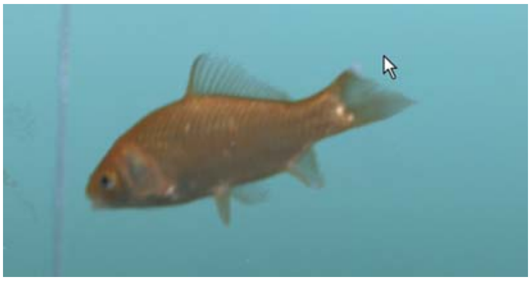
10. Change the keying **operation** to **Clean FG Noise**. This time, sample areas of gray pixels inside the goldfish.

One or two small samples should be enough. The color pick pushes the selected pixels to the opaque part of the matte.

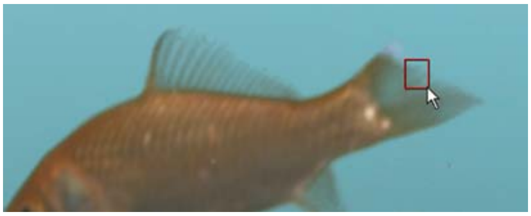


You want to keep the gray pixels inside the fins to retain a semi-transparent matte in these areas. If you go too far, you can always press the undo button in the control panel to step back to the previous action.

11. Press **A** again over the Viewer to toggle to all color channels. Your image should look similar to the example shown below. You may see some detail dropping out from the fins.



12. Change the keying **operation** to **Restore Detail**, and scrub over the fins to bring back some of the edge detail.



You may get different results than those shown here, depending on the pixel values you sample from the image.

Use Restore Detail to push the selected pixels back toward the opaque part of the matte. Use the Make FG Transparent operation to fine-tune the semi-transparent area.

You could go back and forth, between cleaning the background and foreground, but this usually produces a matte with "crunchy" edges. The goal is to find the balance between foreground and background that produces an acceptable matte for your subject.

Later in this chapter, you'll use the rotoscoping tools to clean-up this matte and combine this with the image from the next example.

Image-based Keying

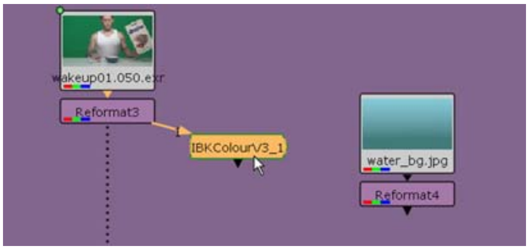
Many keying tools, like Primatte, use a color-pick as the baseline for the matte extraction process and then require the artist to tweak the matte from that baseline. Nuke's image-based keyer (IBK) uses the pixel values of the compositing images, instead of a color-pick, to generate the best matte for the image you want to extract. It works by generating a processed screen image that preserves the color variations of the blue- or greenscreen and using this - rather than a single color - to pull the key. This generally gives good results and speeds up the keying process when working with uneven blue- or greenscreens.

Oaties
Oaties

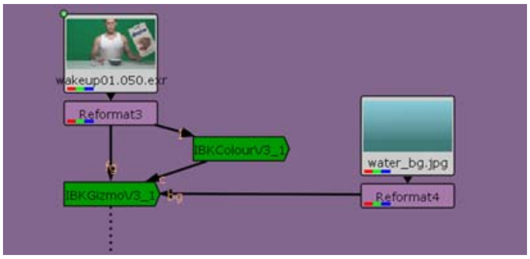
Image-based keying requires two nodes in Nuke. First, you insert an IBKColour node to process the screen image, which is preset to work with either greenscreen or bluescreen. This node generates the processed screen image that preserves the color variations in your blue- or greenscreen. Then, you insert an IBKGizmo node to generate the matte using the processed screen image, the original image, and also the background image for the composite.

To pull a key with IBK:

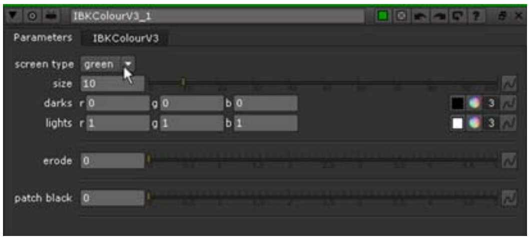
1. In the **keying_tutor.nk** project file, locate the node tree labelled, "**Image-based Keying**".
2. Right-click over the **Reformat3** node and choose **Keyer > IBKColour** from the pop-up menu. Drag the **IBKcolorV3_1** node to the right.


wakeup01.050.exr
Reformat3
IBKColourV3_1
water_bg.jpg
Reformat4

3. Click an empty spot in the node graph to deselect all nodes. Then, choose **Keyer > IBKGizmo** from the pop-up menu.
4. From **IBKGizmoV3_01** node, connect **fg** (foreground) to the **Reformat3** node. Connect **c** (color screen) to the **IBKcolorV3_1** node.
5. Connect **bg** from **IBKGizmoV3_1** to the **Reformat4** node, which supplies the background for the comp.
6. Connect the Viewer to the **IBKGizmoV3_1** node, and your node tree should look similar to this:

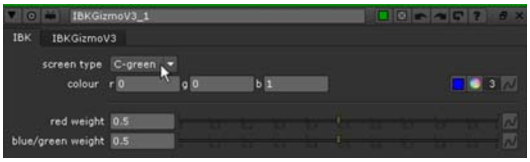

wakeup01.050.exr
Reformat3
IBKColourV3_1
fg
c
IBKGizmoV3_1
bg
water_bg.jpg
Reformat4

7. Open the control panel for **IBKColourV3_1** and change the **screen type** to **green**.


IBKColourV3_1
Parameters
IBKColourV3

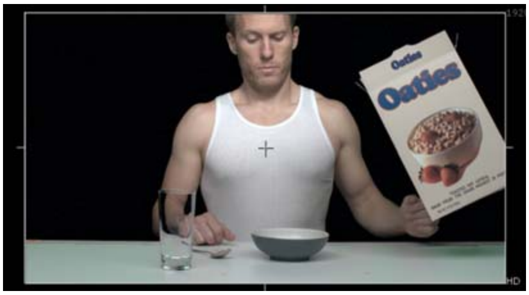
screen type green
size 10
darks r 0 g 0 b 0
lights r 1 g 1 b 1
erode 0
patch black 0

8. Open the control panel for **IBKGizmoV3_1**, and change its **screen type** to **C-green**.


IBKGizmoV3_1
IBK
IBKGizmoV3

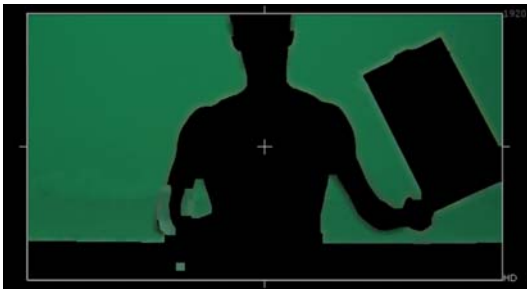
screen type C-green
colour r 0 g 0 b 1
red weight 0.5
blue/green weight 0.5

You should see an acceptable matte, shown in the screen capture below, on frame 50.

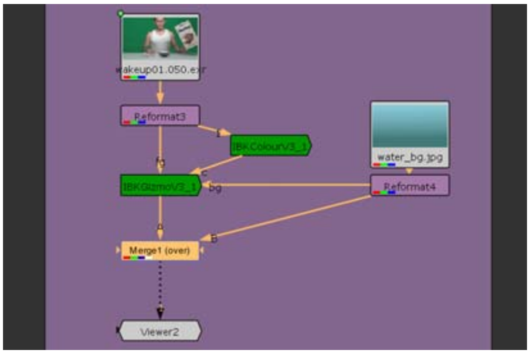


This is a very good start for this image.

9. Connect the Viewer to the **IBKColourV3_1** node. You'll see the processed screen image, which is essentially a Gaussian-filtered high-contrast key.



10. Choose **Merge > Merge** (or press **M** over the node graph) to insert a **Merge (over)** node.

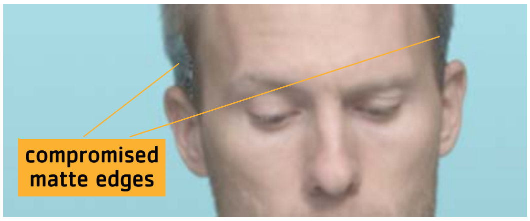

wakeup01.050.exr
Reformat3
IBKColourV3_1
water_bg.jpg
IBKGizmoV3_1
bg
Reformat4
A
B
Merge1 (over)
Viewer2

11. Connect **IBKGizmoV3_1** to the **A** input of the **Merge (over)** node. Then connect the **B** input to the **Reformat4** node.

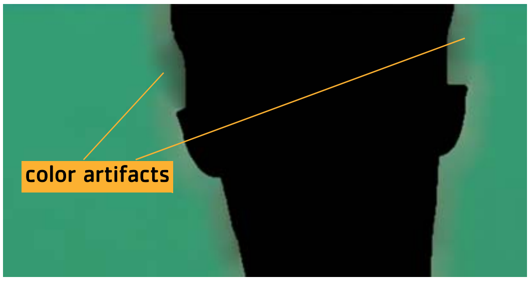
The color of this greenscreen is completely out of the region of the acceptable industry standard, but IBK does a good job anyway, by smoothing the screen and using the result to essentially create a difference matte with the foreground.

Tip *IBK has presets for green and blue screens, but you can also do a color-pick for any screen color inside the IBKGizmo node.*

If you zoom-in on the image, you'll see small areas near the subject's hair, where the matte is compromised.

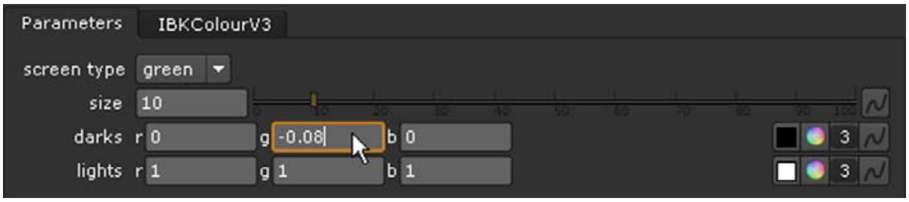


12. Connect the Viewer to **IBKcolorV3_1** and you'll see color artifacts around the edges of the matte.



When you look at the smoothed screen produced by IBKColor, you should see only values of your screen color and black.

13. In the **IBKcolorV3_1** control panel, lower the **darks**, **g** (green) setting to **-0.08**. (If you were keying a bluescreen image, you would lower the "b" value for "darks.").

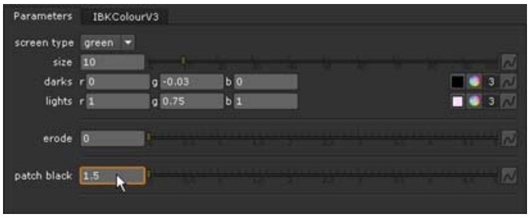


screen type green
size 10
darks r 0 g -0.08 b 0
lights r 1 g 1 b 1

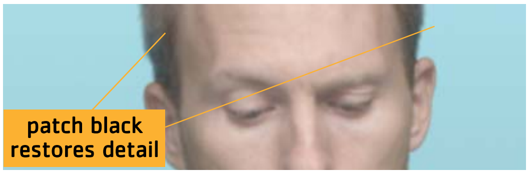
This fixes most of the problems at the hairline, but destroys the good key for the lower portion of the image.

Tip *The artifacts in the IBKColor image appear as specific color shades: light green, dark green, light red, dark red, light blue, and dark blue. To remove these, simply adjust the appropriate controls for the artifacts you want to remove: lights/g, darks/g, lights/r, darks/r, lights/b, and darks/b.*

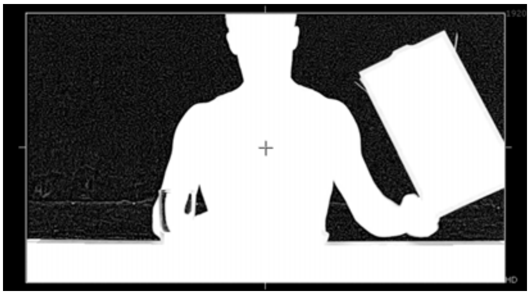
14. In **IBKcolorV3_1**, raise the **darks**, **g** value to **-0.03**. Then change the **lights**, **g** value to **0.75**. This corrects the artifacts of the screen image.
15. Now, change the **patch black** setting to **1.5** to restore the edge detail of the hairline.



screen type green
size 10
darks r 0 g -0.03 b 0
lights r 1 g 0.75 b 1
erode 0
patch black 1.5

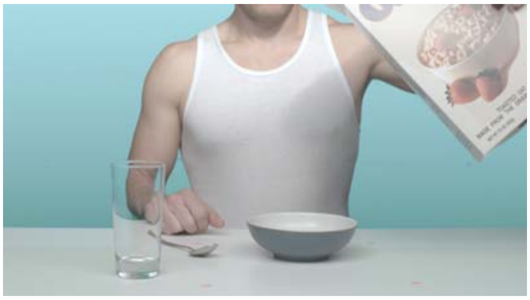


16. Connect the Viewer to the **IBKGizmoV3_1** node. Press **A** and you'll see the current alpha channel produced by the IBK system.



The displayed alpha image shown is correct for the IBK. If the intensity of the noise in your alpha channel is greater than the example show above, you may need to adjust—in very small increments—the **dark** and **light** values for the color channels in the IBKColor node.

17. Press **A** again over the Viewer to toggle back to display all color channels, and scrub through the timeline to check the matte at various points in the clip.



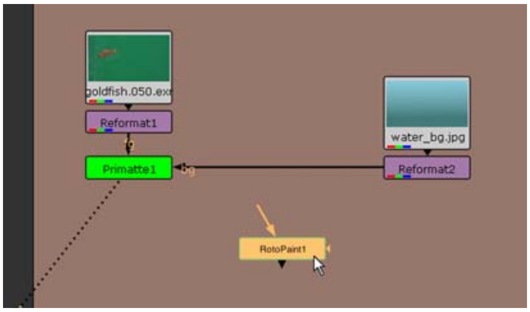
18. If you haven't already done so, save your project under a new file name to save the changes you've made to project.

Rotoscoping

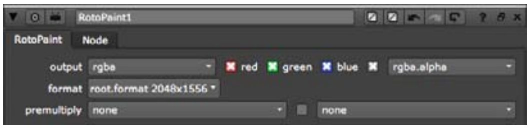
In this example, we'll return to our first keying example to apply a garbage matte and clean-up the aquarium image.

To draw a garbage matte:

1. Go back to the node tree from the first example, and connect the Viewer to the **Primatte1** node. Drag the time slider to frame **50**.
2. Click an empty spot on the node graph to deselect all nodes, and choose **Draw > RotoPaint** from the pop-up menu.

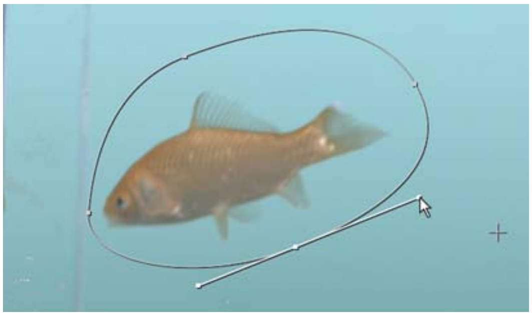


3. At this point, you don't need to connect the **RotoPaint1** node to anything, but its control panel must be open, and the first tab, **RotoPaint**, should be active.



output rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha
format root.format 2048x1556
premultiply none ☐ none

4. Inside the Viewer, you'll see the goldfish image. Click the **Bezier tool** in the RotoPaint toolbar on the left side of the Viewer and in the Viewer, click four points around the goldfish to create a rotoshape. You can drag+click to draw a point and adjust its curve at the same time.



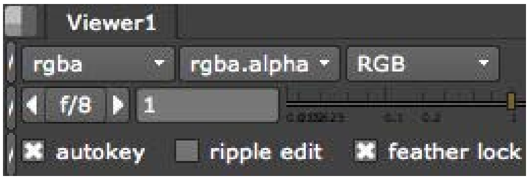
Tip *As long as the RotoPaint1 control panel is open, you can view and edit the rotoshape. You can press* ***O*** *over the Viewer to toggle the display overlay, if necessary. Click the right mouse button over any point to select options for the rotoshape.*

Because this will be a garbage mask, we want to edit the shape to remove elements from the glass aquarium.

5. Drag the points and adjust the tangents—the handles on each of the points—to refine the rotoshape.

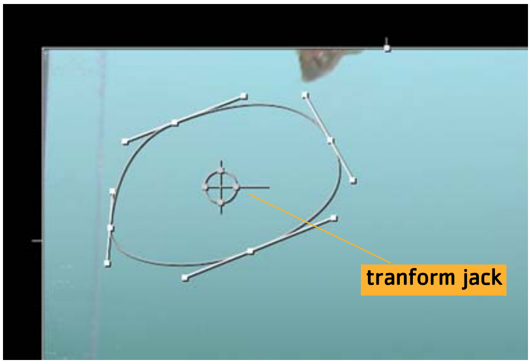
Now we need to animate the garbage matte to follow the motion of the fish.

6. In the RotoPaint tool settings panel, on top of the Viewer, the **autokey** option should be active. If not, click the box for this option.

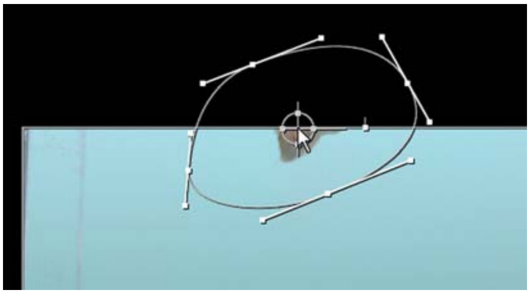


☒ autokey ☐ ripple edit ☒ feather lock

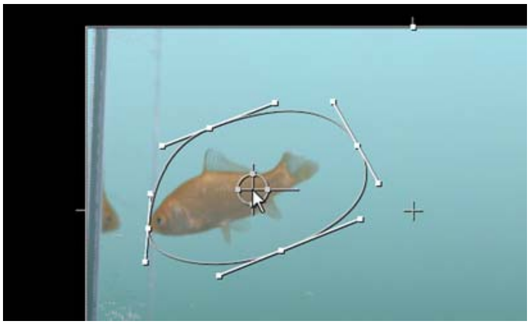
7. Move the time slider to frame **1** and click the **Transform** tab in the RotoPaint control panel. Then select the entire Bezier shape in the stroke/shape list (at the bottom of the control panel) or by clicking one of the points in the shape using the **Select All** tool. A transform jack appears.



8. Drag the center point of the transform jack, and move it over the current position of the goldfish.



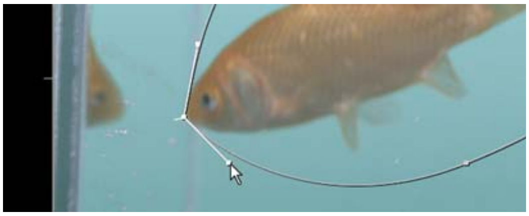
9. Go to end of the timeline, to frame **60**. Drag the shape once more to adjust for the movement of the goldfish.



If your Bezier shape is similar to the one shown above, then you probably don't need more than the three key frames at frames 1, 50, and 60.

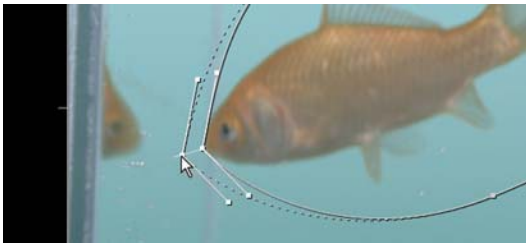
However, you may want to scrub through the timeline and make adjustments.

10. Scrub to frame **60** on the timeline and you'll see the roto gets a little close to corner-line that we want to remove from the aquarium glass.
11. Click on an empty spot in the Viewer to deselect all points. Then, press **Ctrl/Cmd** and click on the point near the goldfish's nose, to temporarily break the point's tangent handle.
12. Adjust the handles to create a peak at the fish's nose.



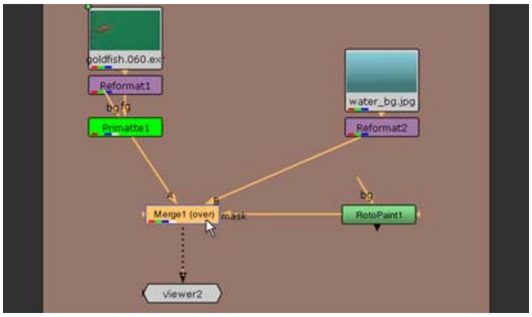
Now, for good measure, let's create a feathered edge for this particular point.

13. With the point selected, drag the feather handle away from the fish to create a feathered edge for this point, at this frame.

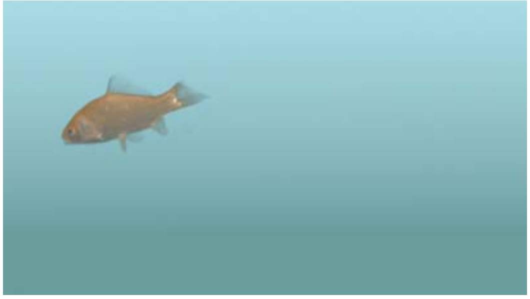


So you've drawn and animated the rotoshape. Let's wire it into the node tree to mask out the "garbage."

14. Drag the bg connector off the **Primatte1** node to disconnect it from the **Reformat2** node.
15. Choose **Merge > Merge** from the pop-up menu. Connect **Primatte1** to the **A** input on the over node. Connect **Reformat2** to the **B** input.


goldfish.060.exr
Reformat1
bg fg
Primatte1
water_bg.jpg
Reformat2
A
B
bg
Merge1 (over)
mask
RotoPaint1
Viewer2

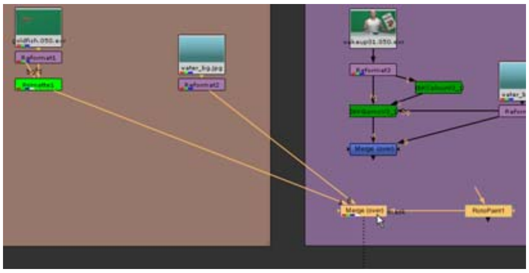
16. Connect the **RotoPaint1** node to the mask connector on the over node. This effectively removes the aquarium reflections in the image.



You might want to scrub through the timeline to see if there are places where you need to adjust the rotoshape.

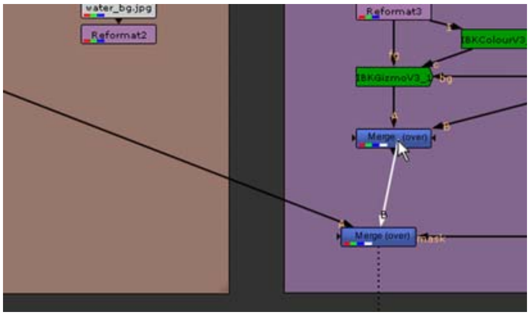
If you want to take this a little further, you can now add the goldfish to the composite from the second example.

17. Select and drag the **Merge (over)** node and the **RotoPaint1** node below the node tree you used for the IBK example.



18. Drag the **Viewer** node over, as well, and keep it connected to the **Merge (over)** node.

19. Drag the **B** connector from the **Reformat2** node and connect it to the over node in the IBK node tree.



The Viewer shows the result. Of course you might want to add a **Transform** node after the first **Merge (over)** node, to size and position the goldfish. Otherwise, this project is completed.

Oaties
Oaties

Keying Video

Nuke's Keyer node provides standard controls for pulling luma keys, green and blue screens, and color channels. We'll use this keyer—and a few other nodes—to handle a special keying situation: video.

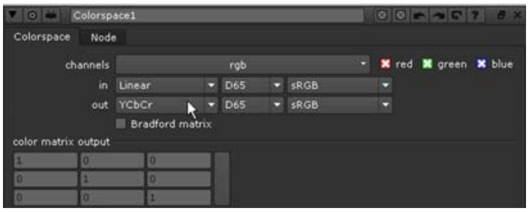
We'll begin by inserting a group of nodes that allow you to pull a cleaner matte by filtering the compression artifacts in the chroma red and chroma blue channels of digital video. This involves converting the image back to its original colorspace, blurring the channels with the artifacts, and then converting the image back to Nuke's native linear colorspace.

To prepare video footage for keying:

1. In the "**Keying Video**" node tree, select the **fgman.0001.dpx** node.

   When Nuke reads images into the workspace, it converts them to a linear colorspace. So here the first step is to convert the video image back to video YCbCr colorspace.

2. Right-click over the node tree and choose **Color > Colorspace** from the pop-up menu.
3. In the **Colorspace1** control panel, change the **out** parameter to **YCbCr**.



Colorspace1

Colorspace | Node

channels rgb ☒ red ☒ green ☒ blue

in Linear D65 sRGB

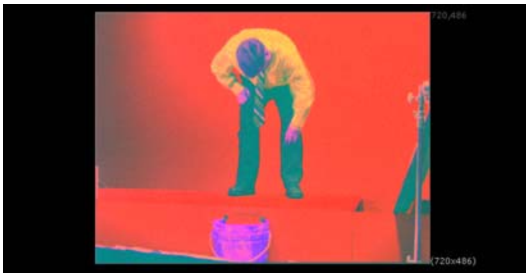
out YCbCr D65 sRGB

☐ Bradford matrix

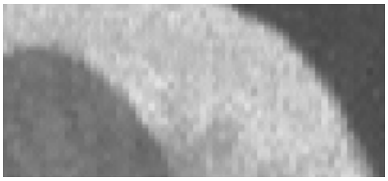
color matrix output

1	0	0
0	1	0
0	0	1

No, you're not having an 80's flashback. What you're experiencing is the result of the red, green, and blue channels remapped to the native video channels for luma (Y), chroma blue (Cb) and chroma red (Cr), respectively.



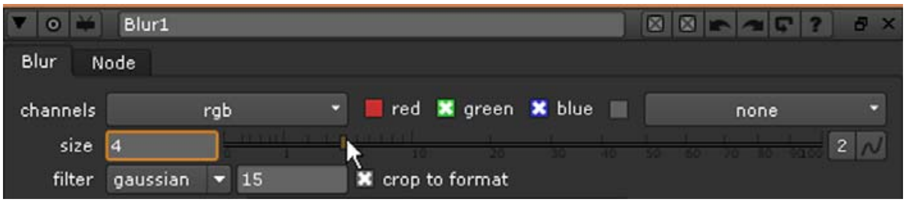
4. Press **r** over the Viewer to look at the Y channel. Press **g** to view Cb, and **b** to view Cr.



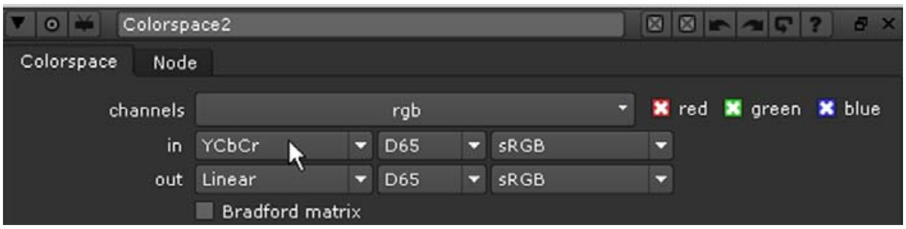
Above, you see the Y channel image on the left. The Cb channel image is shown on the right. Notice the "blocky" compression artifacts in the Cb channel. These make it difficult to get a clean edge for your matte, but since most of your detail is in the Y channel, you can apply a small blur operation to the Cb and Cr channels to improve the situation without losing much detail.

5. Press **r**, **g**, or **b** again to toggle back to all color channels.

6. Right-click on the Colorspace1 node and choose **Filter > Blur**. In the Blur1 control panel, set the blur **size** to **4**.


Blur1
Blur
Node
channels rgb ☐ red ☒ green ☒ blue ☐ none
size 4
filter gaussian 15 ☒ crop to format

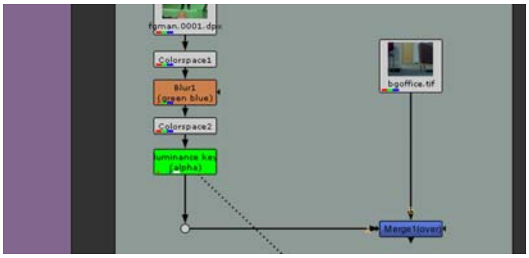
7. Select **rgb** in the **channels** menu and uncheck the **red** channel box. You don't want the blur operation to process the image in the red channel (the remapped Y or luma channel) because this channel is uncompressed.
8. Right-click on the Blur1 node and add another **Color > Colorspace** node. Change the **in** parameter to **YCbCr** and the **out** parameter to **Linear**. This converts the image back to standard rgb/linear.


Colorspace2
Colorspace
Node
channels rgb ☒ red ☒ green ☒ blue
in YCbCr D65 sRGB
out Linear D65 sRGB
☐ Bradford matrix

To pull a basic greenscreen key:

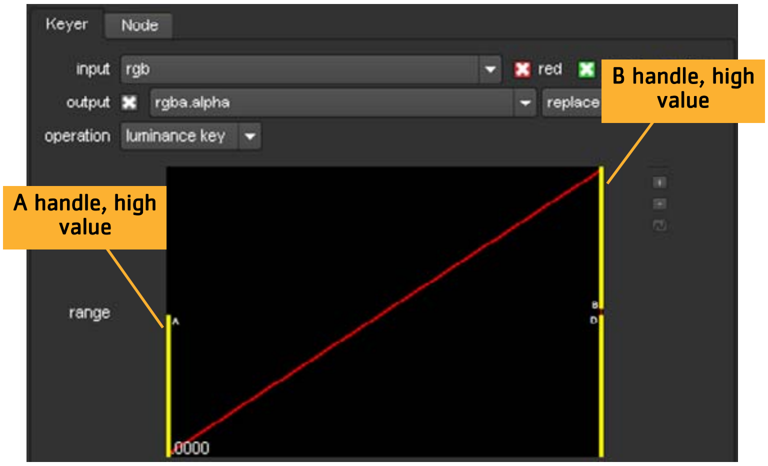
1. Select the **Colorspace2** node, and choose **Keyer > Keyer** from the right-click menu.

   This inserts a keyer named "luminance key," which is the default keying operation for this node.


fgman.0001.dpx
Colorspace1
Blur1
(green blue)
Colorspace2
luminance key
(alpha)
bgoffice.tif
Merge1(over)

2. Attach a Viewer to the **luminance key** node and then press **A** to display the alpha channel.

In the control panel for the Keyer node, you'll see the "range" graph:

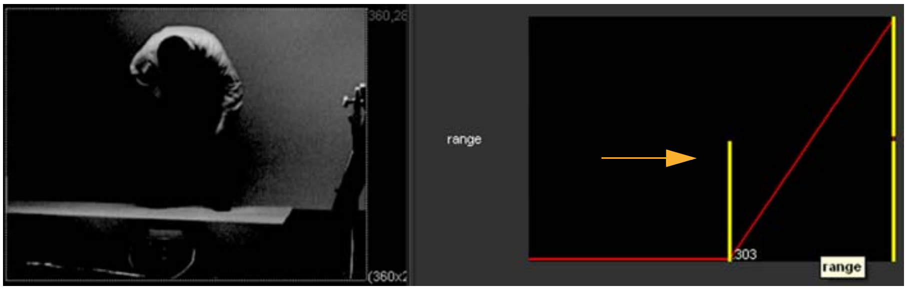


input rgb
output rgba.alpha
operation luminance key

Keyer
Node
red
replace
B handle, high value
A handle, high value
range
A
B
D
0000

The range graph is where you'll adjust the low and high pixel values of the matte. The first yellow handle on the left determines the low or transparent values of the key and second handle, on the upper-right, determines your high or opaque values.

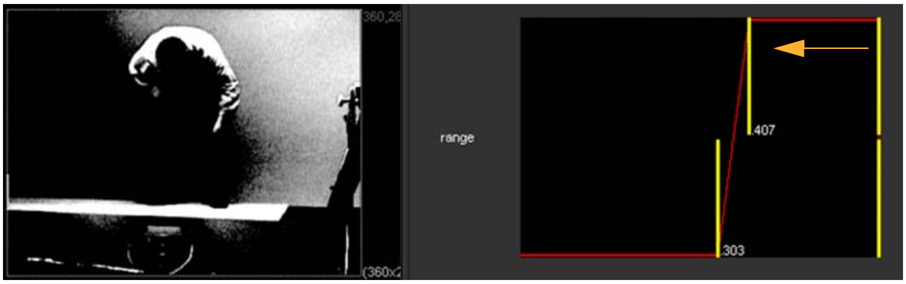
3. Drag the first yellow handle to the right until it reads **.303** (approximately), and watch the effect in the Viewer.



360,28
(360x2
range
303
range

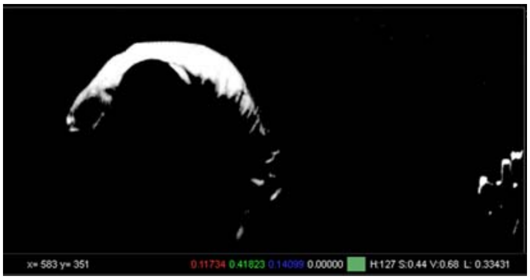
This sets the low value for the matte. Any pixels that fall below this value are clipped to black.

4. Drag the yellow handle, located at the upper-right edge, to the left until it reads **.455** (approximately).

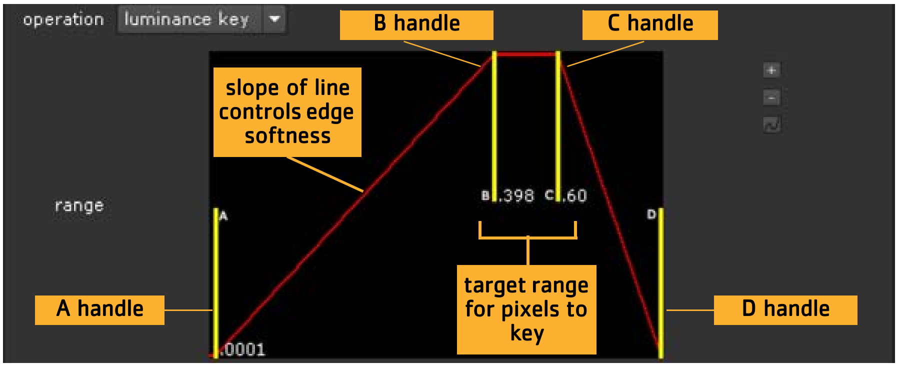

360,28
(360x2
range
407
303

This sets the high value for the matte. Pixel values above this setting are clipped to white. At these settings, it's not quite "matte-worthy," so let's make an adjustment.

5. Drag the **A** handle to change the low setting from **.303** to **.424**, and drag the **B** handle to change the high value from **.455** to **.61**.

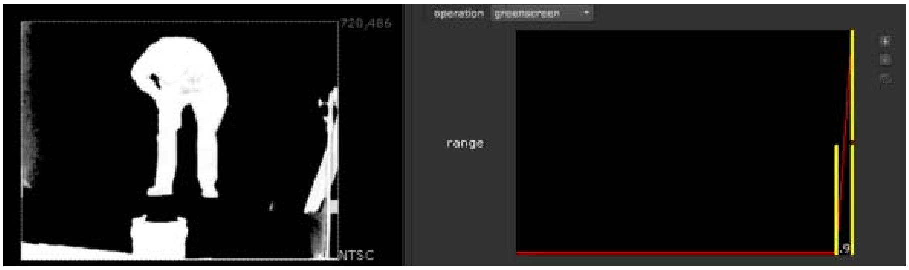

x= 583 y= 351
0.11734 0.41823 0.14099 0.00000
H:127 S:0.44 V:0.68 L: 0.33431

As you adjust the location of the handles, the slope of the line controls the softness, or level of grays, for the matte edge. A gradual slope produces a softer edge. A sharper slope produces a jagged or crunchy edge—drag the **A** handle on top of the **B** handle and you'll see what that means.


operation
luminance key
B handle
C handle
slope of line controls edge softness
range
A
B .398
C .60
D
A handle
.0001
target range for pixels to key
D handle

The default positions let you control the low and high values, assuming your image has distinct light and dark areas. However, sometimes the subject of the matte falls into the "middle-gray" area; the **C** and **D** handles on the curve, after the first two, let you shift the center for the high values of the key.

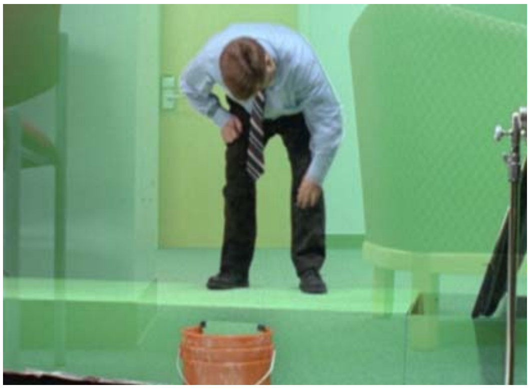
6. Change the keying **operation** to **greenscreen**. Pull the high value handle to the right, up to the value of 1.0. Then, set the low value at **.90**.



operation greenscreen

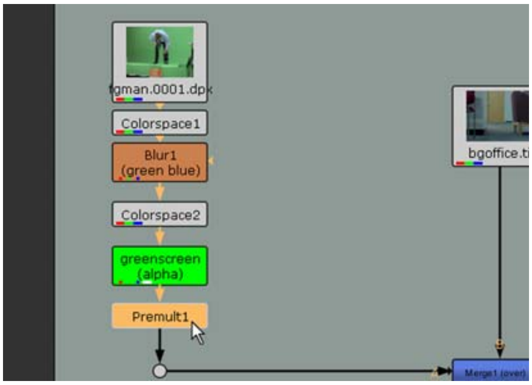
There's a lot of garbage around the image, but it looks like you've got a fairly clean edge around your subject. Let's check it in the comp.

7. Switch the Viewer to display all color channels and then attach the Viewer to the **Merge (over)** node.

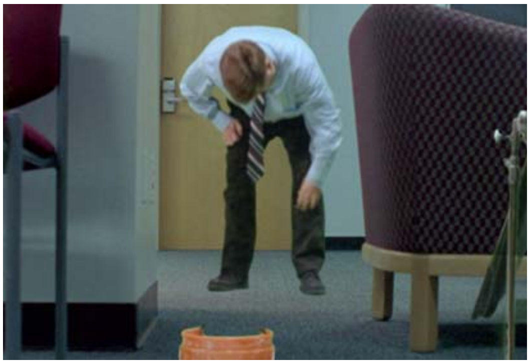


That's completely terrible. What happened? The alpha channel created by the greenscreen keyer must be multiplied into the pixel values of the original image in order to generate the matte. Some keyers, like Primatte, provide a "composite" output option which handles the multiplication for you. The Keyer node does not, so you'll have to do it manually.

8. Select the **greenscreen** keyer node and choose **Merge > Premult** from the right-click menu.

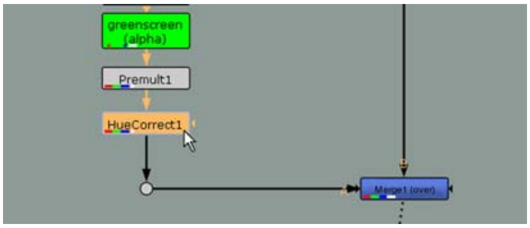

fgman.0001.dpx
Colorspace1
Blur1
(green blue)
Colorspace2
greenscreen
(alpha)
Premult1
bgoffice.ti
Merge1 (over)

9. Adjust the high and low ranges values in the greenscreen node's control panel to refine the edges around your subject.



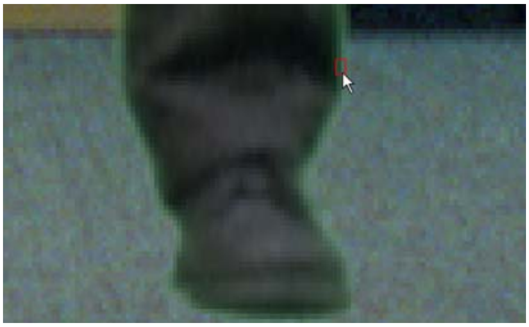
This keyer isn't particularly good at handling spill, so let's add a color-correction node to remove those green edges.

10. Select the **Premult1** node and then choose **Color > HueCorrect** from the right-click menu.


greenscreen
(alpha)
Premult1
HueCorrect1
Merge1 (over)

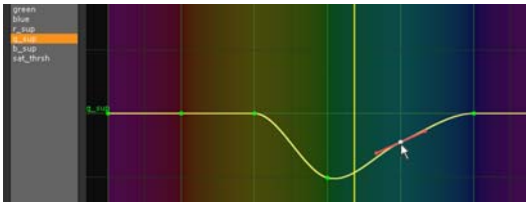
11. Select the **g_sup** parameter in the **HueCorrect1** control panel to select the green suppression curve.

12. Over the Viewer, press the **Ctrl** or **Command** key and scrub over the green edges.



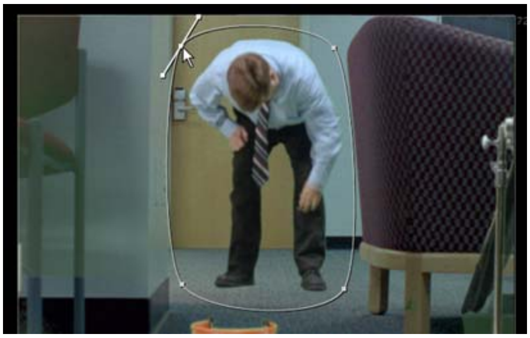
Inside the HueCorrect1 control panel, you'll see a yellow vertical line, which marks the place in the curve that you need to lower to suppress the samples pixels.

13. While viewing the edges you want to suppress, adjust the **g_sup** curve so that it looks similar to this:

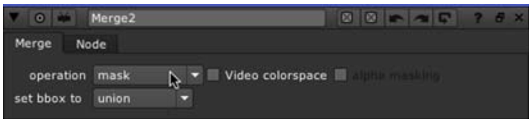


When you're satisfied with the spill-suppression, you may want to add a quick garbage matte to remove the rigging.

14. Click on an empty place in the node tree, and add a **Draw > RotoPaint** node.
15. Select the **Bezier** tool in the RotoPaint toolbar on the left side of the Viewer and click (and drag if you want curved points) over the Viewer to draw a Bezier shape around the man. Four points should be enough.



16. Select the RotoPaint node and press **M** to add a Merge node— that's the shortcut for choosing **Merge** > **Merge** from the menu—and change the merge **operation** to **mask**.



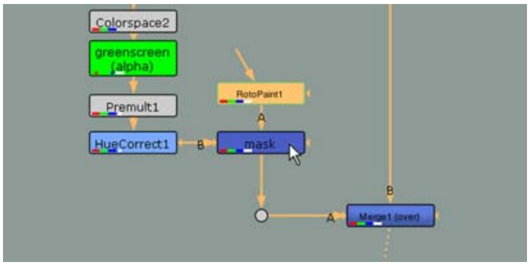
Merge2

Merge | Node

operation: mask ☐ Video colorspace ☐ alpha masking

set bbox to: union

17. Rewire the nodes as shown below. The **RotoPaint1** node should be connected to the A input on the mask node, **HueCorrect1** should be connected to the B input. Connect the output of **mask** to the "elbow" dot.



This method of masking is a little different than what you did in the previous example. The point here is that there are different ways to structure these types of composites. Your results should look similar to the screen capture below.

Epilog

Keying is rarely a simple matter of picking the screen color you want to remove. To get the very best mattes, you often need to combine several techniques and you've learned several in this chapter. You've pulled mattes with Primatte and Nuke's Image-based Keyer, and you've used the rotoscoping tools to cleanup a matte and control the parts of the image you want to use in the composite.

You've also seen how to key video footage by converting the image to its native colorspace and filtering the compressed channel data to pull a cleaner matte.

- End of Tutorial -

TUTORIAL 4: 3D INTEGRATION

Nuke's 3D workspace creates a powerful compositing environment within your project script. This workspace combines the advantages of cameras, lighting, and a three-axis (x, y, and z) environment, with the speed of node-based compositing. You pipe 2D images into the 3D space, setup a camera, animate your scene, and then render the results back to the 2D composite.

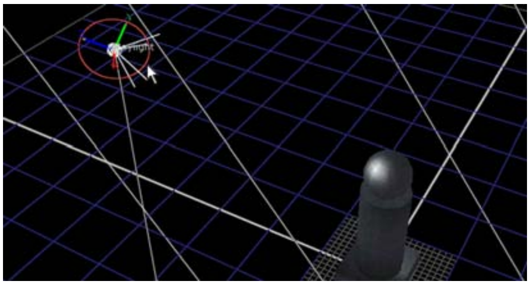


Figure 4.1: Compositing in the 3D workspace.

The Basic 3D System

The 3D workspace is defined by a group of nodes in your script. The most basic setup includes a Camera node, a Render node, a Scene or Geometry node, and nodes that provide the 2D images you want to pipe into the 3D compositing space.

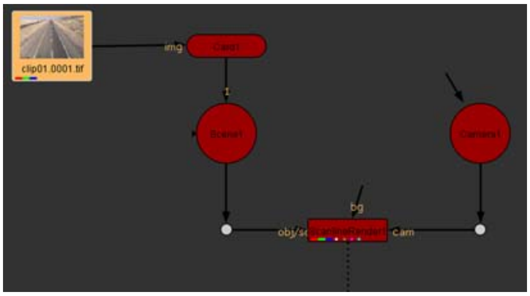


Figure 4.2: The basic 3D node tree: 2D image, geometry, scene, render, and camera.

The 3D Viewer

Once you have the 3D node structure, you can use any Viewer in Nuke as a gateway to the 3D compositing space. Choose **3D** from the view list, or press the **Tab** key over the Viewer to toggle between the 2D and 3D views.

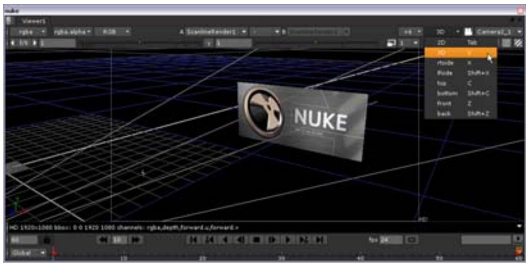


Figure 4.3: Each Viewer window can also display the 3D workspace.

On the view list, you'll also see orthographic views—rtside, lfside, top, bottom, front, back— which provide non-perspective views into the scene. In three-quarter perspective, it can be difficult to accurately place objects on the axes, and the non-perspective views make it easier to line things up.

The Geometry or Scene Node

Every 3D system needs a piece of geometry—a card, a sphere, a cube, something— to receive an image or clip that the camera can "see." One is all you need, but you can setup complex systems with a large amount of 3D data. When you have two or more objects for a 3D system, you need a Scene node to create a "place" where the camera (and the ScanlineRender node) can see all the objects at once.

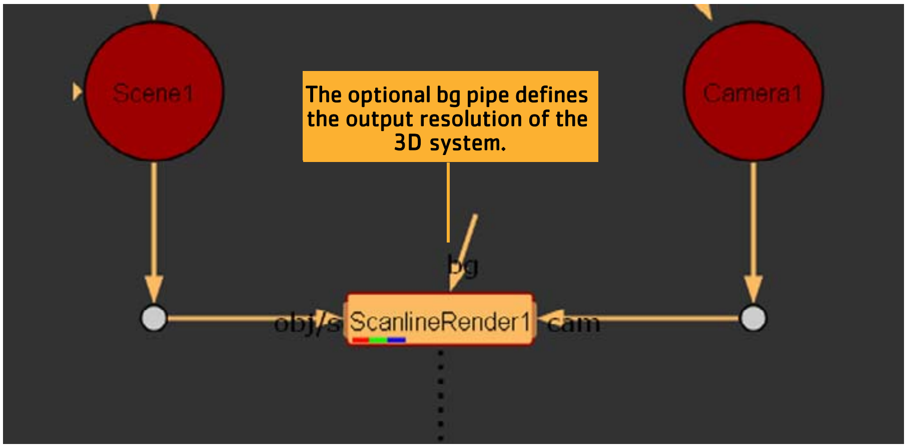
The Camera Node

The Camera node creates your view into a scene. It has several controls to help you match the properties of a physical camera. You can animate its position or import animation or tracking data to matchmove your 3D scene with a background plate. A 3D system can have multiple cameras connected to the Scene node, to create different views on a 3D scene.

Tip *Only one camera can be connected to the ScanlineRender node to generate the output, but you can insert multiple ScanlineRender/Camera node pairs to generate output from various perspectives.*

The ScanlineRender Node

The last node, ScanlineRender, sends the results of your 3D scene back into your composite as a 2D image. It's always 2D in, 3D manipulation, and then 2D back out, which is why this is often called "2-and-a-half-D."



The optional bg pipe defines the output resolution of the 3D system.

Figure 4.4: The scanline render node converts the image back to 2D.

The image created by the ScanlineRender node will be the same resolution as your project settings. When you need to render a specific resolution, use the optional "bg" pipe. Connect a Constant node with the resolution you want and that will define the output of the ScanlineRender node.

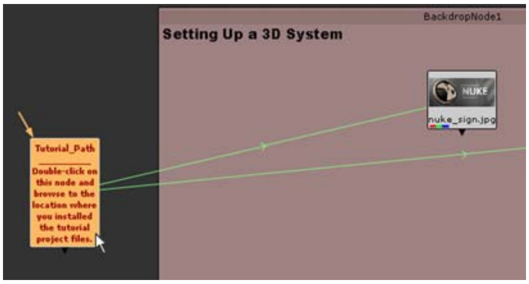
So now you know the basic 3D setup for your compositing script. Let's take a test drive.

Open the Tutorial Project File

The "3Dinteg_tutor.nk" project file includes the node trees for the first part of this chapter.

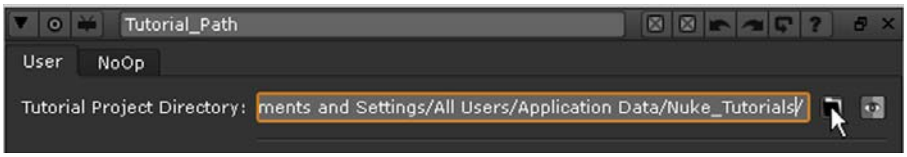
To open the project file:

1. Launch the Nuke application and choose **File > Open** from the menu bar.
2. In the file browser, navigate to your **Nuke_Tutorials/3DInteg/** folder, select the **3dinteg_tutor.nk** project file and click **Open**.
3. Locate the **Tutorial_Path** node, on the left side of the script, and double-click it to open its control panel.



Double-click on this node and browse to the location where you installed the tutorial project files.

4. Click the "file folder" button. Browse to the location where you installed the tutorial project files, and then click **Open** to select the location.



Tutorial Project Directory: ments and Settings/All Users/Application Data/Nuke_Tutorials/

   After you select the correct path, the error messages should clear from the Read nodes, and the thumbnails in the script will update with the correct images.
5. Close the **Tutorial_Path** control panel. Then, choose **File > Save As** to save a copy of the project file.
6. Move the mouse pointer over the node graph, and press **F** to frame the entire contents of the project file.

   The green arrows (lines) show the links between the Tutorial_Path node and the Read nodes.
7. If you wish, press **Alt+E** to hide the expression arrows.

The Tutorial_Path node saves the location of the project files on your computer, so you don't need to repeat this for future sessions with this project file.

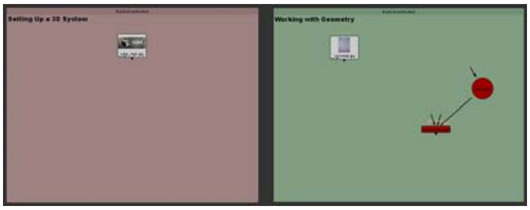


Figure 4.5: Node trees in the 3dinteg_tutor.nk project file.

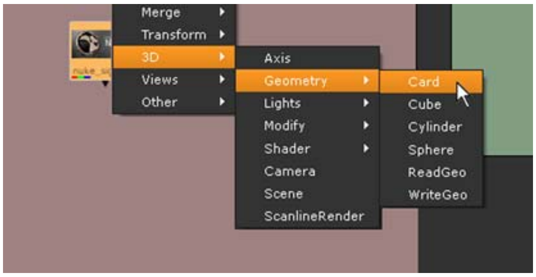
Setting Up a 3D System

Let's start with the basics. In this first example, you'll create a basic 3D node tree, map an image to a 3D card, manipulate it, and then render the result back out to the 2D composite.

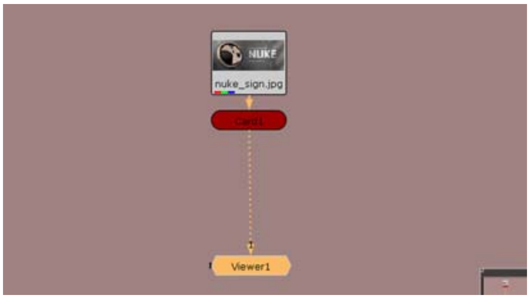
To setup a 3D node tree:

1. In the "3Dinteg_tutor.nk" project file, locate the backdrop node labeled "**Setting Up a 3D System**." You'll see a Read node with the image you'll use for this example.

2. Right-click over the **nuke_sign.jpg** node, and choose **3D > Geometry > Card**.

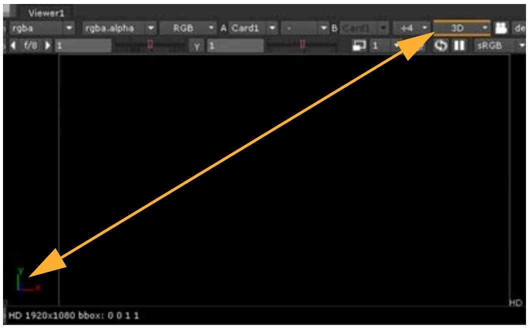

Merge
Transform
3D
Views
Other
Axis
Geometry
Lights
Modify
Shader
Camera
Scene
ScanlineRender
Card
Cube
Cylinder
Sphere
ReadGeo
WriteGeo

This attaches a "Card1" node. Let's see what it looks like in 3D.


NUKE
nuke_sign.jpg
Card1
Viewer1

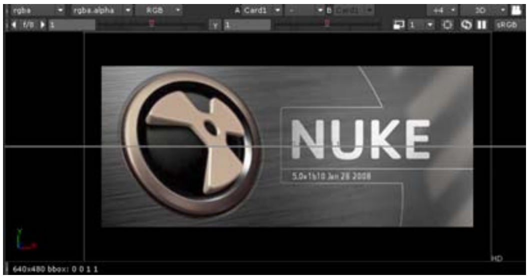
3. Attach a Viewer to the **Card1** node and Nuke switches the Viewer to 3D.

   Wow, that's amazing. It looks exactly like the 2D Viewer. How can anyone tell the difference? Check the lower-left corner of the Viewer and you'll see an orientation marker for the three axes in 3D. You'll also see that "3D" is displayed on the view list.


Viewer1
rgba
rgba.alpha
RGB
A Card1
B
3D
sRGB
HD 1920x1080 bbox: 0 0 1 1
HD

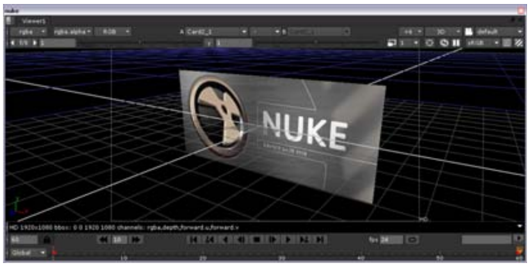
That sign is a little darker than expected, isn't it? Actually, you can't see the image yet because the default view of the 3D workspace is at the origin or center of the space. Perhaps zooming-out will improve the view.

4. Press the **Alt** key (Windows /Linux) or the **Option** key (OS X), and drag with the middle mouse button to zoom or "dolly." Drag to the left and you'll zoom out.



Hey, look. There's the Nuke emblem. In the 3D Viewer, the "pan" and "zoom" controls are exactly the same as what you've used for the node tree and the 2D Viewer, but let's try "tumbling" to get a better view.

5. **Alt-** or **Option**-drag with the right mouse button to rotate around the origin point of the 3D workspace. You now see the 3D grid and the image mapped to the card.



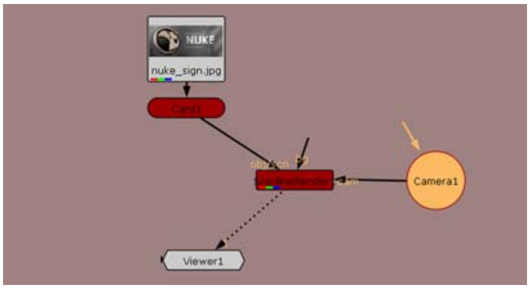
When an image is connected directly to a Card node like this, it is applied as a flat or "planar" map. The size of the card adjusts to the dimensions of the image.

6. Click on the card and you will select the node in the node tree and also the card inside the 3D workspace.
7. Use the mouse (and the **Alt** or **Option** key) to navigate through the workspace. Go ahead, pan, dolly, and rotate at will. Then, press **F** over the Viewer to frame the 3D view.

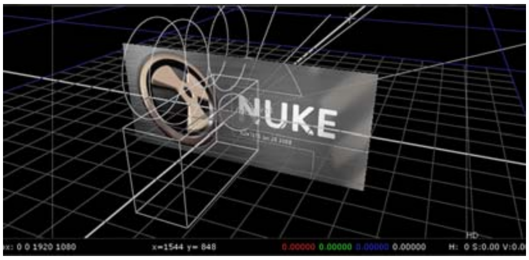
Tip *If you don't like the standard navigation controls, open the **Preferences** control panel (**Shift+S**), select the **Viewers** tab and change the **3D control type** to **Maya**, **Lightwave**, or **Houdini**.*

8. Click on an empty spot in the Node Graph to deselect all nodes. Let's add the other nodes you need.

9. Right-click on the Node Graph and choose **3D > Camera** from the pop-up menu. Keep its control panel open so you can manipulate the camera in the Viewer.
10. Right-click and choose **3D > ScanlineRender** to insert a render node, and then connect the nodes as shown below.

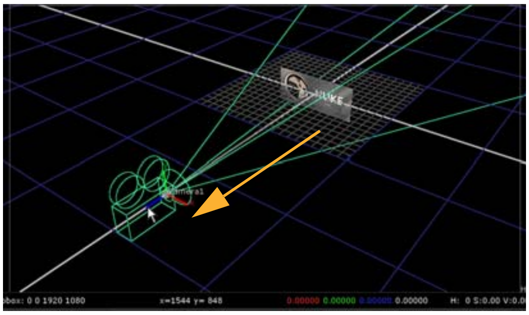


11. Connect the Viewer to the **ScanlineRender** node, and you have the most basic 3D system in Nuke.
12. Press **Tab** over the Viewer to change to the 2D view. You won't see the Nuke emblem—hey, where did it go? We saw it before.
13. Press **Tab** again to switch back to 3D. You'll see the default camera position is too close to view the card. Let's move things around to get an image for 2D.

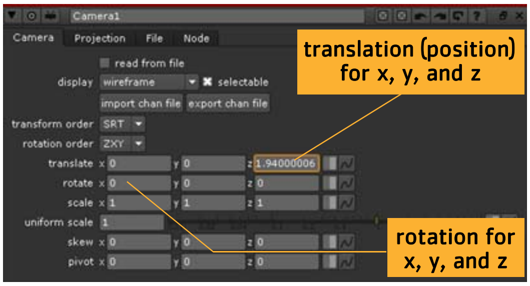


To position objects in the scene:

1. **Alt-** or **Option-**drag with the middle mouse button to dolly out and show more of the 3D workspace.
2. Select the camera. You can do this by clicking the camera object in the Viewer or clicking the **Camera1** node in the Node Graph.
3. Drag the transform handles to move the camera away from the card, along the z-axis.


NUKE
Camera1
bbox: 0 0 1920 1080
x=1544 y= 848
0.00000 0.00000 0.00000 0.00000
H: 0 S:0.00 V:0.0

As you drag the camera, look at the camera's control panel. You'll see the x/y/z transform values reflect the camera's current position.

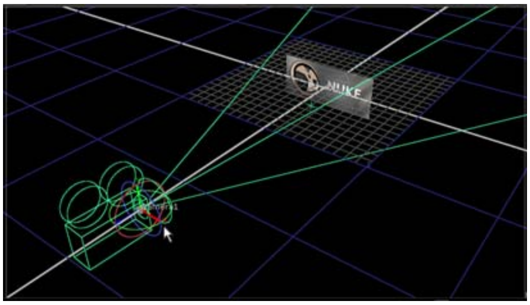

Camera1
Camera
Projection
File
Node
import chan file
export chan file

☐ read from file
display wireframe ☒ selectable
transform order SRT
rotation order ZXY
translate x 0 y 0 z 1.94000006
rotate x 0 y 0 z 0
scale x 1 y 1 z 1
uniform scale 1
skew x 0 y 0 z 0
pivot x 0 y 0 z 0

translation (position) for x, y, and z

rotation for x, y, and z

4. Press **Ctrl** (Mac users press **Command**) over the Viewer and the transform handles change to rotation rings.



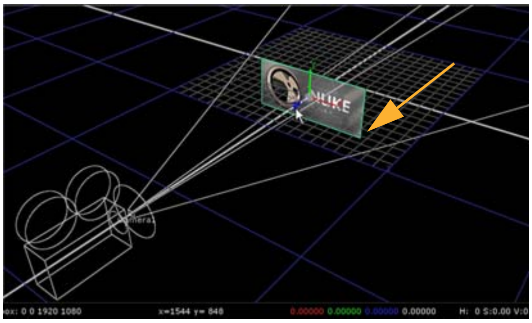
5. Drag the green ring to rotate the camera around the Y-axis. Notice the x/y/z rotation values in the control panel reflect the angle of the rotation.

   The blue handle "rolls" or rotates on the Z-axis, and the red handle rotates on X.

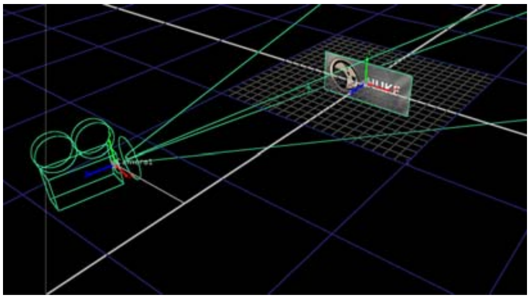
6. Now, select the card object and move it away from the camera.

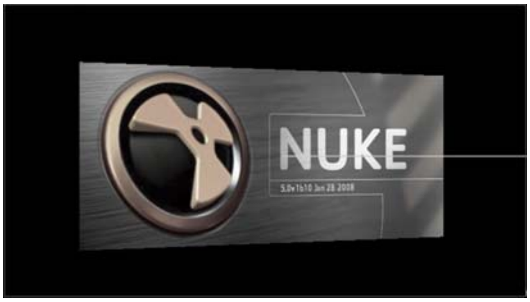
   Keep the control panel open for the Card node. As with the Camera node, the transform handles disappear from the Viewer when you close the control panel.

7. Drag the card's transform handles to position it in the 3D workspace. If you wish, press the **Ctrl** key (Mac users press **Command**) over the Viewer and rotate the card.



8. Press **Tab** over the Viewer to switch between the 2D and 3D views to see the image the ScanlineRender node will produce.





9. Before you continue to the next example, close all the control panels that are currently open.

   In this example, it doesn't matter where you move the camera or card. In reality, however, you often need to use specific values, which you can enter directly in the control panels.

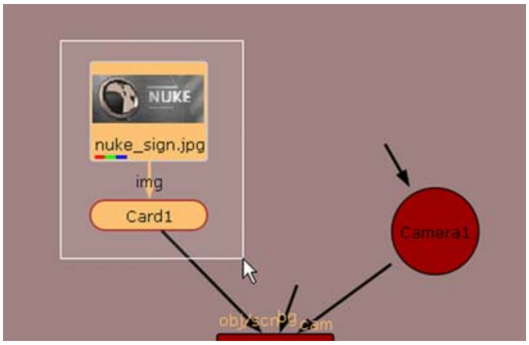
   You can also import camera data or animation curves—did you notice the **import chan file** button in the camera's control panel?—and apply them to the objects in the workspace.

Making a Scene

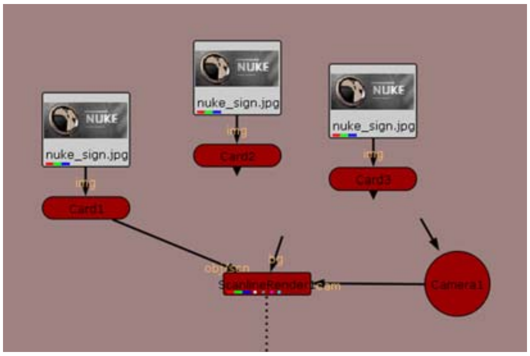
We mentioned earlier the Scene node creates a place where multiple objects may be seen by the camera and the render node. If you only have a single object, you don't need a Scene node, but where's the fun in that? Scene nodes make it possible to really tap into Nuke's ability to handle huge amounts of 3D information, and you should know how to use it.

To setup a scene:

1. Inside the "Setting Up a 3D System" node tree, drag a selection around the **nuke_sign.jpg** node and the **Card1** node to select them.

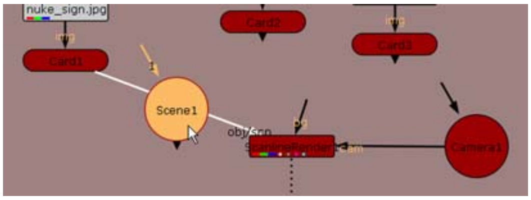

NUKE
nuke_sign.jpg
img
Card1
Camera1
obj/scn
bg
cam

2. Press **Ctrl+C** (Mac users press **Command+C**) to copy the selected nodes or choose **Edit > Copy** from the pop-up menu.
3. Press **Ctrl+V** (Mac users press **Command+V**) or choose **Edit > Paste** to insert a copy of the nodes. Press **Ctrl+V** or **Command+V** again to insert a second copy, and then arrange the nodes as shown below.

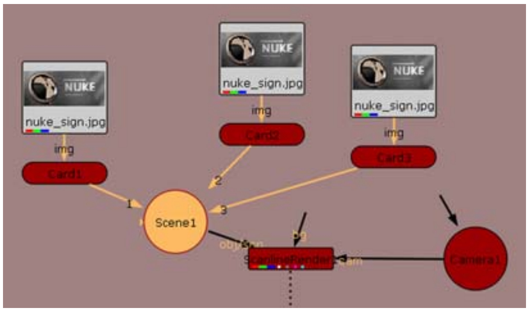

NUKE
nuke_sign.jpg
img
Card1
NUKE
nuke_sign.jpg
img
Card2
NUKE
nuke_sign.jpg
img
Card3
obj/scn
bg
ScanlineRender1
cam
Camera1

4. There are multiple cards now, and you need a Scene node to create a space where the rendering node can see all cards at once.
5. Click on an empty space in the node graph to deselect all nodes. Right-click and choose **3D > Scene** to insert the "Scene1" node.

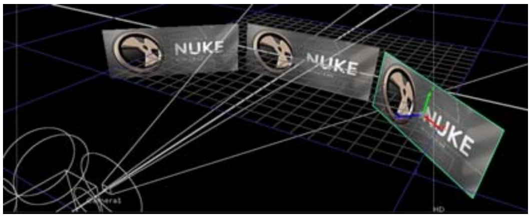
6. Drag the **Scene1** node onto the obj/scn connector to insert the Scene1 node between **Card1** and **ScanlineRender1**.


nuke_sign.jpg
img
Card1
Card2
img
Card3
1
Scene1
obj/scn
bg
ScanlineRender1
cam
Camera1

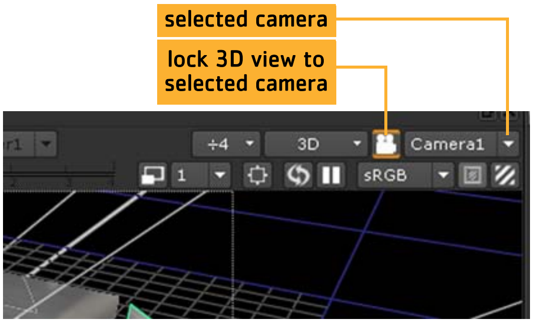
7. Connect the **obj/scn** connector from the **ScanlineRender1** node to the **Scene1** node. Connect each **Card** node to the **Scene1** node.


NUKE
nuke_sign.jpg
img
Card1
Card2
Card3
1
2
3
Scene1
obj/scn
bg
ScanlineRender1
cam
Camera1

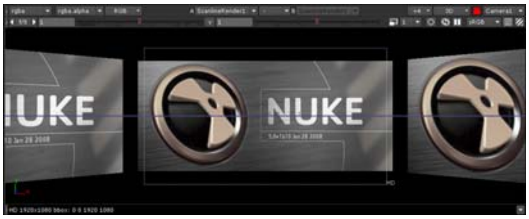
8. Double-click on the **Card1** node to open its control panel. In the Viewer, you'll see the transform handles for the first card.
9. Move and rotate the card to a different position. For this example, it doesn't matter where you place it.
10. Open the control panel for **Card2** and move its card to a different place in the scene.
11. Open the **Card3** control panel and move that card, also.


NUKE
NUKE
NUKE

You could switch to the 2D view to see the result, but why not just look through the camera? Next to the view list, you see the button that locks the 3D view to the camera.



12. From the view list, choose **3D** (**V**) to switch to a 3D perspective view. Then click the "lock view to 3D camera" button.

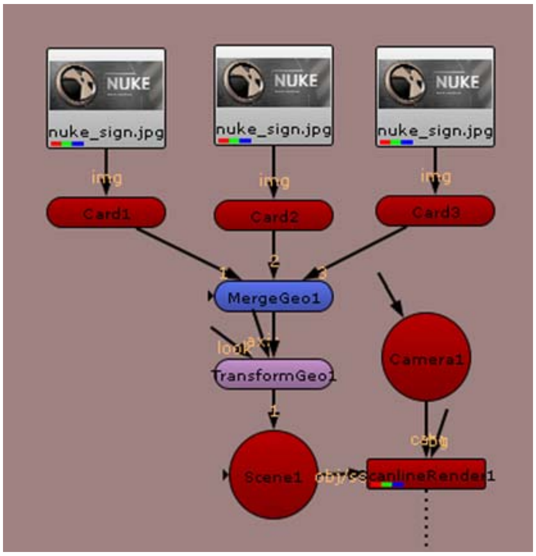


13. Turn off the "lock 3D view to camera" button. You won't need it during the rest of this tutorial.

Merging and Constraining Objects

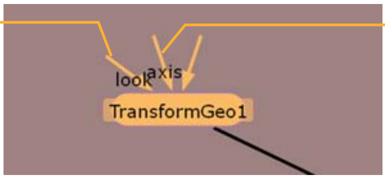
You can merge objects and move them together as a group. To do so, you need to insert MergeGeo and TransformGeo nodes after the objects. The MergeGeo node first merges the objects together, after which you can use the controls of the TransformGeo node to move the merged objects in the 3D space. You can also use the TransformGeo node to constrain objects, as you will notice later in this tutorial.

To merge the three card objects together, right-click on the Card1 node and select **3D** > **Modify** > **MergeGeo**. This inserts a MergeGeo node between Card1 and Scene1. Disconnect the Card2 and Card3 nodes from the Scene node and connect them into the MergeGeo node. Then, right-click on the MergeGeo node and select **3D** > **Modify** > **TransformGeo**. Your node tree should now look like the following:


NUKE
nuke_sign.jpg
nuke_sign.jpg
nuke_sign.jpg
img
img
img
Card1
Card2
Card3
1
2
3
MergeGeo1
look
axis
TransformGeo1
Camera1
Scene1
obj/scn
cam
ScanlineRender1

On the TransformGeo nodes, you see multiple connectors. The connector without a label should be attached to a geometry object or a MergeGeo node. The other connectors act as constraints on the connected object's position.

You can connect "look" to the object or camera that the current object should "face."


look
axis
TransformGeo1

You can connect "axis" to an Axis node. This links the position of the current object to the position of the Axis node.

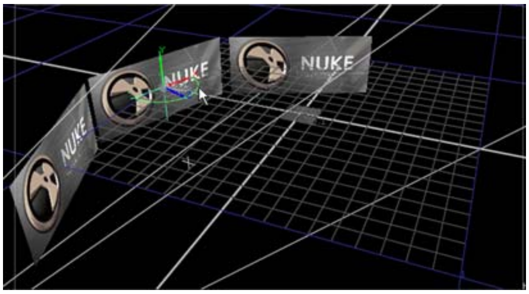
When a camera or object is connected to the optional **look** connector, the TransformGeo node adjusts the rotation so that the object's z-axis always "points" to the camera or object.

The **axis** connector can be used to link the current object to the position, rotation, and scale of a special 3D object called the Axis node. If you've worked with other 3D applications, you know the Axis node as a "null" or "locator" object.

You are still working with the "Setting Up a 3D System" node tree. The following steps show how you can move the merged nodes, and also how to make objects "look" at the camera and other objects.

To move the merged objects together:

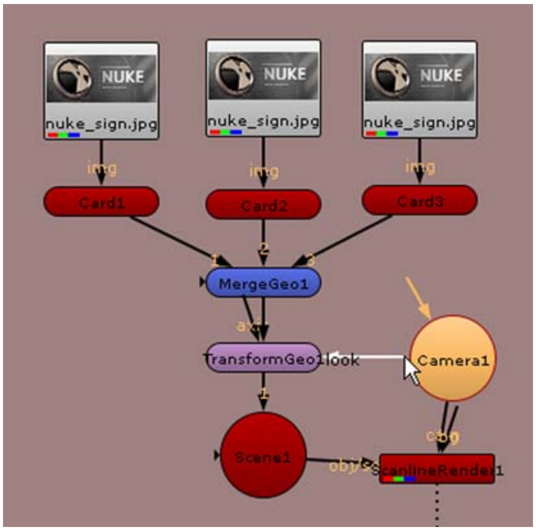
1. Click on the TransformGeo1 node to select it. Its control panel should also be open and you'll see its transform handles in the Viewer.
2. Drag the handles to move all the cards merged with the MergeGeo node.
3. Press the **Ctrl** or **Command** key and drag the rings to rotate the cards as a group.


NUKE
NUKE
NUKE

4. In the **TransformGeo1** control panel, drag the **uniform scale** slider to increase the size of the entire group of cards.

To make objects 'look' at the camera:

1. Drag the **look** connector from the **TransformGeo1** node onto the **Camera1** node.


NUKE
nuke_sign.jpg
NUKE
nuke_sign.jpg
NUKE
nuke_sign.jpg
img
img
img
Card1
Card2
Card3
1
2
3
MergeGeo1
axis
TransformGeo1
look
Camera1
cam
Scene1
obj/scn
ScanlineRender1

In the Viewer, you'll now see the TransformGeo1 node is constrained to the location of the camera.

2. Select and move **Camera1** in the Viewer window. As you do so, the three cards controlled by the **TransformGeo1** node rotate to "look at" the camera location.

   Why is this useful? Let's assume you have a 2D matte painting mapped to a card in your scene. The "look" option ensures that the plane of the painting always faces the camera, regardless of the camera position, and maintains the illusion depicted by the painting.

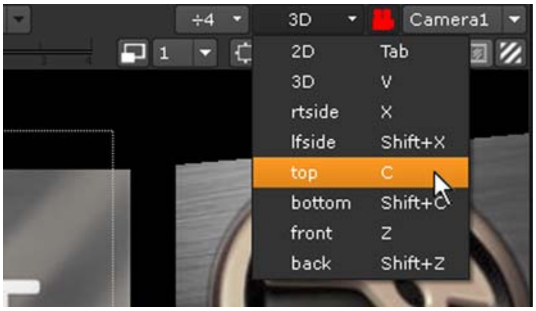
Before you move on, disconnect the TransformGeo node's **look** connector from the camera.

Animating a Scene

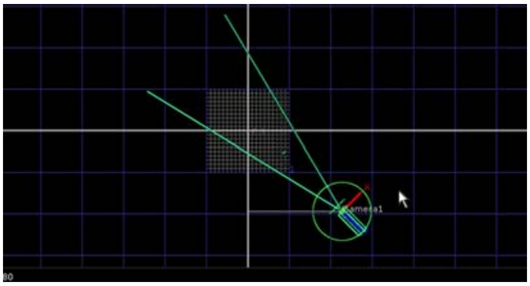
The little scene you've created would be more interesting with a camera move. You can animate both cameras and objects; in each control panel, you'll see an Animation button next to each parameter you can animate over time.

To animate the camera:

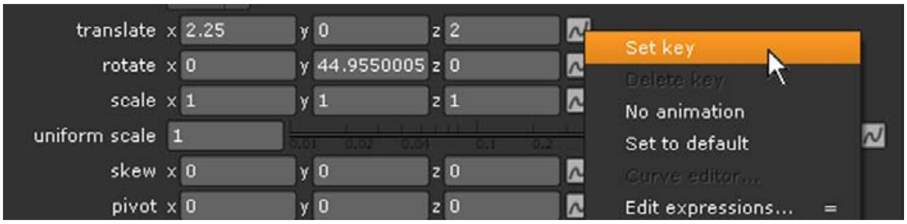
1. In the Viewer, drag the time slider to frame **1** on the timeline.
2. Let's switch to an overhead view to move the camera. Choose **top** from the view list.



3. Double-click on the **Camera1** object (either inside the Viewer or on the node graph) to open its control panel.
4. Move the camera to the right and rotate it to "look" at the center of the 3D workspace.



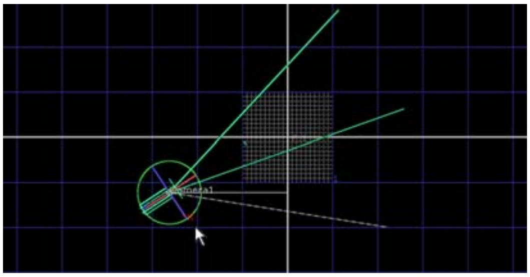
5. Click on the animation button next to the camera's **translate** parameters, and choose **Set key**.



translate x 2.25 y 0 z 2
rotate x 0 y 44.9550005 z 0
scale x 1 y 1 z 1
uniform scale 1
skew x 0 y 0 z 0
pivot x 0 y 0 z 0

The background of the parameter boxes change color to show that a keyframe is now set for these values at the current frame in the timeline. Now, you need to set a keyframe for the rotation values.

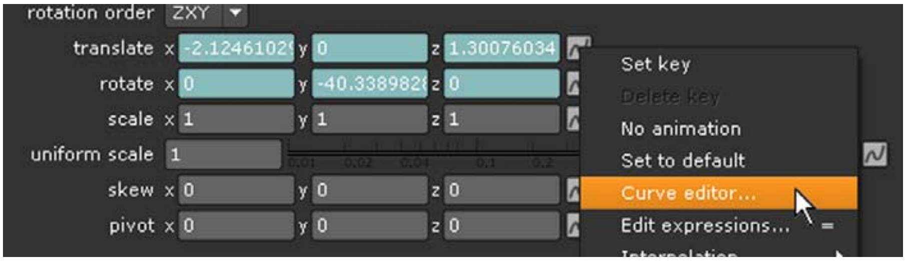
6. Next, click the animation button next to the camera's **rotate** parameters, and choose **Set key**.
7. In the Viewer, scrub to the end of the timeline. Then, move the camera to the left side and rotate it to face center. This automatically sets keys for the **translate** and **rotate** parameters, for the last frame.



8. Drag through the timeline and the camera moves between the positions recorded by key frames.

   Yawn. With only two key frames, the camera moves in a straight line, from start to finish. Let's edit the animation curves to make this more interesting.

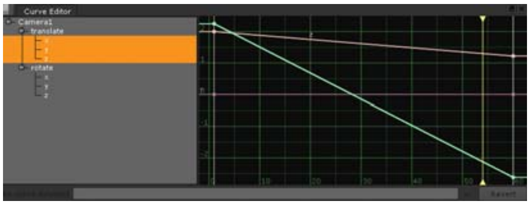
9. Click the animation button next to the camera's translate parameters and choose **Curve editor** from the pop-up menu.



rotation order ZXY
translate x -2.1246102 y 0 z 1.30076034
rotate x 0 y -40.338982 z 0
scale x 1 y 1 z 1
uniform scale 1
skew x 0 y 0 z 0
pivot x 0 y 0 z 0

Set key
Delete key
No animation
Set to default
Curve editor...
Edit expressions...
Interpolation

This opens the Curve Editor window in the same pane as the Node Graph. The outline at the left side of the Curve Editor displays a list of the parameters you've animated, and the curves show the values plotted over time. Each dot shows the value recorded for a value on a keyframe.

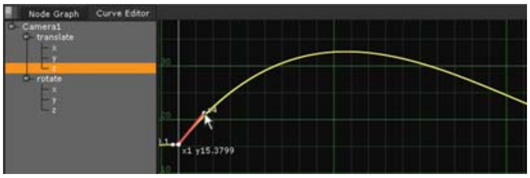


Curve Editor
Camera1
translate
x
y
z
rotate
x
y
z
10
20
30
40
50

Yes, we know. They don't look like curves - yet. You only set two key frames, remember? You can press **Ctrl+Alt** (Mac users press **Command+Option**) and click on any line to add more key frames. However, let's assume you want to create a smooth arc between the first and last recorded position of the camera. Rather than set more key frames, let's just change the shape of the curve.

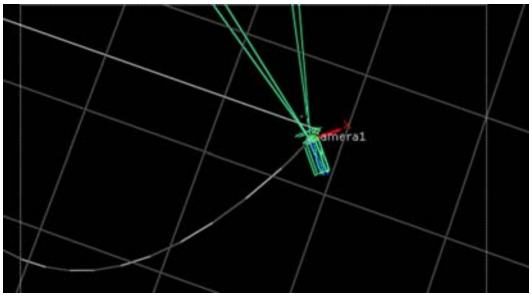
10. You want to control the shape of the camera path along the z—the distance between the origin point and the camera, so click the **translate / z** parameter in the Curve Editor.

    Click on the first point of the translate/z curve to select it, and drag the tangent handle upward, as shown below.

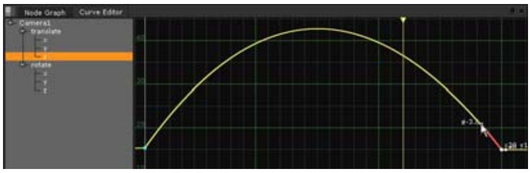


Node Graph
Curve Editor
Camera1
translate
x
y
z
rotate
x
y
z
x1 y15.3799

From this point forward, the curve increases the distance of the camera on the z-axis, which you'll now see in the Viewer.

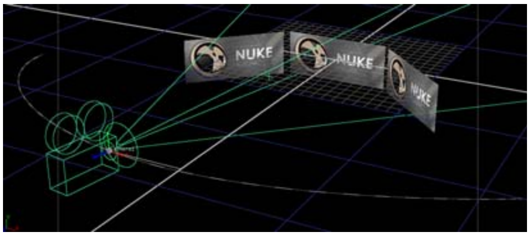


11. Click on the last point of the translate/z curve to select it, and drag it upward also to finish the desired shape. This eases the distance back toward the value of the keyframe at the end of the timeline.



Select the camera in the Viewer and you should see the gradual slopes of the curve create a more interesting arc for the camera move.

Switch to the **3D** (**V**) perspective view and scrub through the timeline to see the new camera move.



Your version may look a little different than this, depending on the positions and rotations you defined, but you get the idea.

12. If you wish, you can set additional key frames to refine the camera path. Hold down **Ctrl+Alt** or **Command+Option** and click on the z-axis curve in the Curve Editor, to add new points to the curve, and then adjust their positions.
13. Before you continue, click the **Node Graph** tab to hide the Curve Editor and return to the node trees.
14. Close all the control panels that are currently open.

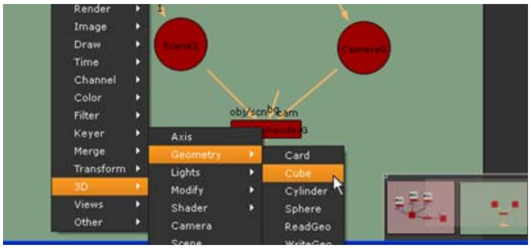
Working with Geometry

In the previous example, you worked with the card object. Nuke also includes primitive geometry, which can be used as set-extension geometry or placeholders for other elements you plan to add to scene.

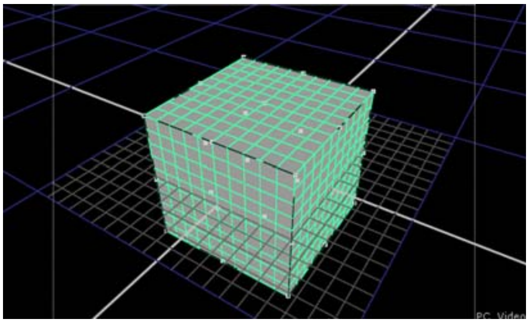
To add primitive objects to the scene:

1. In the "3Dinteg_tutor.nk" project file, locate the node tree labelled "**Working with Geometry**."

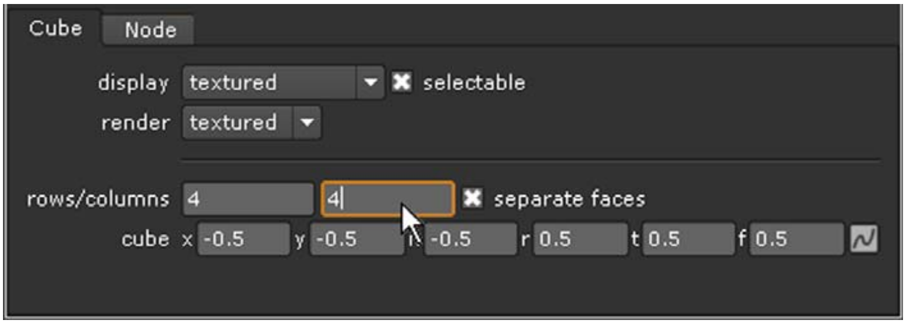
   We've already supplied the 3D node tree with a camera for you, so you need to add the geometry objects, and also create a "scene" where they can coexist.
2. Right-click over the node graph and choose **3D > Scene**. Connect **Scene2** to **ScanlineRenderG**.
3. Connect a Viewer to the **ScanlineRenderG** node and switch to the **3D** perspective view.
4. Right-click and choose **3D > Geometry > Cube**.


Render
Image
Draw
Time
Channel
Color
Filter
Keyer
Merge
Transform
3D
Views
Other
Axis
Geometry
Lights
Modify
Shader
Camera
Scene
Card
Cube
Cylinder
Sphere
ReadGeo
WriteGeo
obj/scn
bg
cam

The default cube primitive appears at the center of the 3D workspace. Let's reduce the number of subdivisions on the cube.

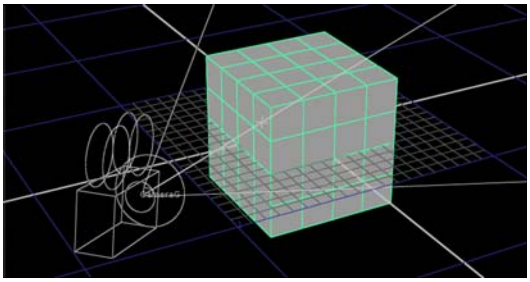


5. In the Cube1 control panel, change the **rows** parameter to **4**. Change the **columns** parameter to **4**, also.

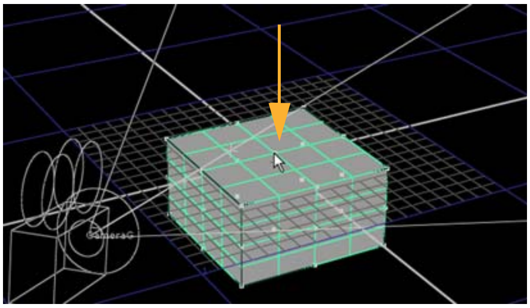


display textured ☒ selectable
render textured
rows/columns 4 4 ☒ separate faces
cube x -0.5 y -0.5 n -0.5 r 0.5 t 0.5 f 0.5

6. Connect the **Cube1** node to the **Scene2** node. Now let's adjust the shape of the cube.



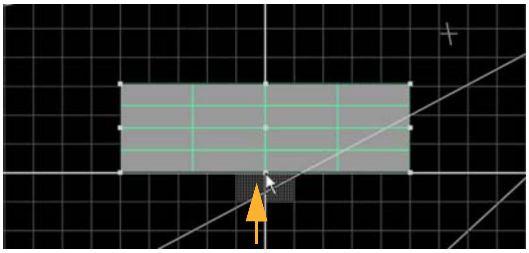
7. Reduce the height of the cube by dragging the top-center point down.



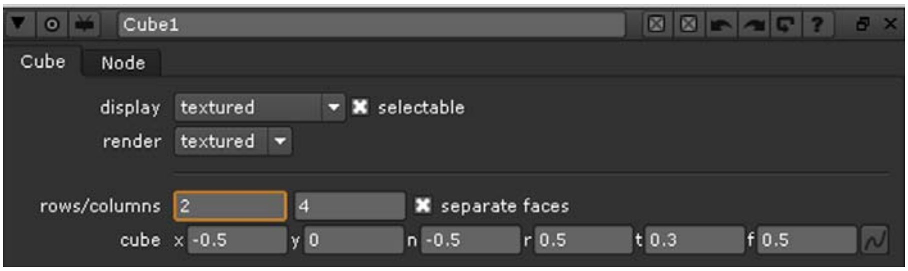
8. From the view list, choose the **front** view to see a non-perspective view of the cube. These non-perspective views can help you size and position objects with more accuracy than you might get in a perspective view.

   Mm... the cube is actually below the x-axis. Let's move it up, but check the values in the **Cube1** control panel.

9. Drag the top of the cube until the **t** (top) value in the **Cube1** control panel is about **0.3**. Drag the bottom of the cube to align it with the x-axis.



10. It looks like you don't need 4 divisions on the sides of the cube, so change the number of rows to **2** in the **Cube1** control panel.



Cube1

display textured ☒ selectable

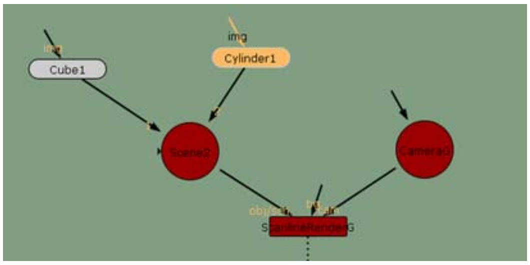
render textured

rows/columns 2 4 ☒ separate faces

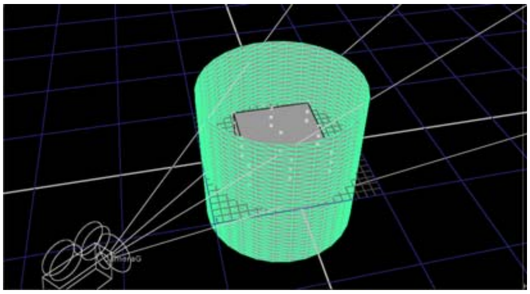
cube x -0.5 y 0 n -0.5 r 0.5 t 0.3 f 0.5

Now let's add a few more primitives—a cylinder and a sphere.

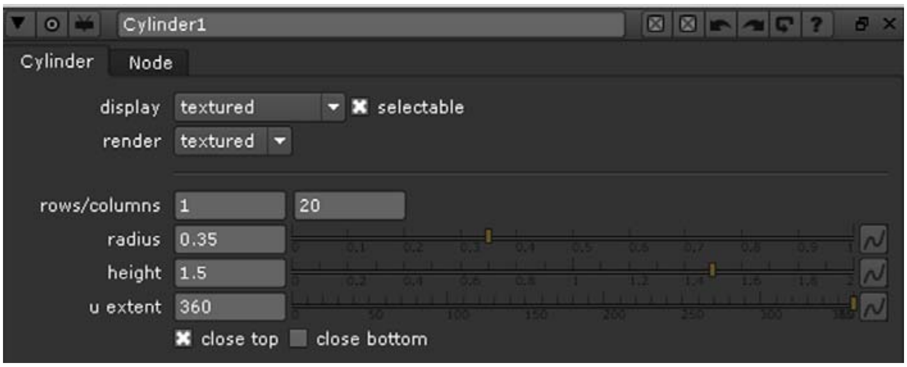
11. Right-click on the node graph and choose **3D > Geometry > Cylinder**. Connect the **Cylinder1** node to the **Scene2** node.



12. Change the view to **3D** (**V**) and zoom out a little to see the whole cylinder.



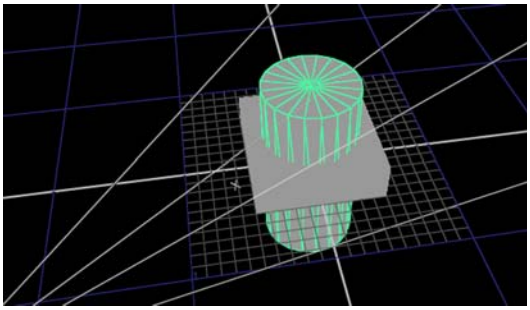
13. In the Cylinder1 control panel, set the **rows** to **1**, the **columns** to **20**.



Cylinder1

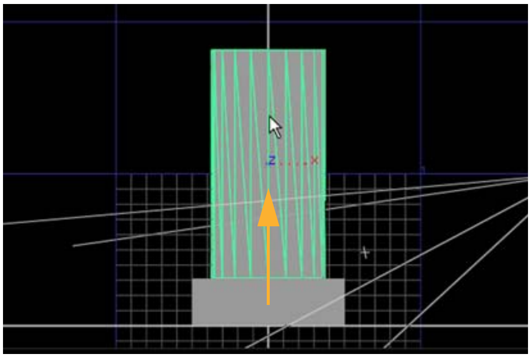
display textured ☒ selectable
render textured
rows/columns 1 20
radius 0.35
height 1.5
u extent 360
☒ close top ☐ close bottom

14. Set the **radius** to **0.35** and the height to **1.5**. Also check the box for **close top**.

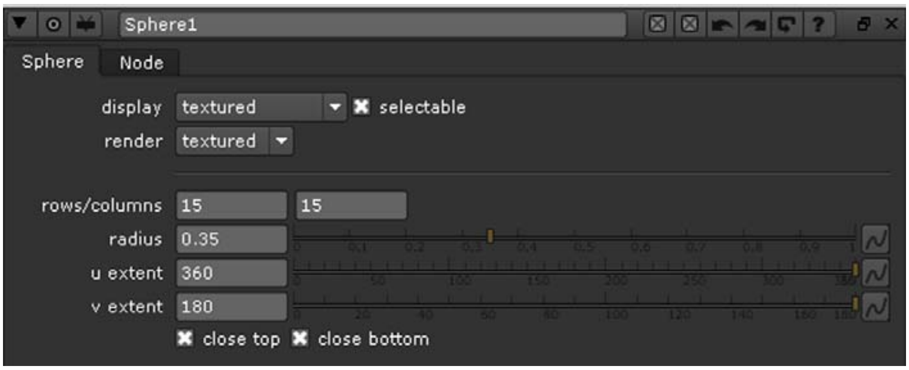


So now you have a cylinder in the scene.

15. Choose **front** from the view list and move the cylinder up to rest on top of the cube.



16. Now add a sphere. Choose **3D > Geometry > Sphere**. In the Sphere1 control panel, set both the **rows** and **columns** to **15**, and change the **radius** to **0.35**.



Sphere1

Sphere | Node

display: textured ☒ selectable

render: textured

rows/columns: 15 15

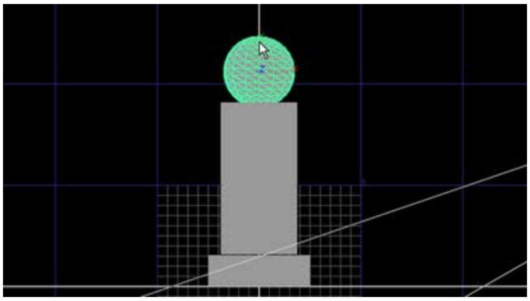
radius: 0.35

u extent: 360

v extent: 180

☒ close top ☒ close bottom

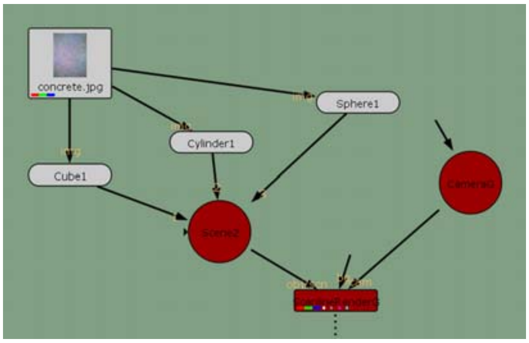
17. Make sure the Sphere control panel is open and move the sphere object to cap the top of the cylinder.

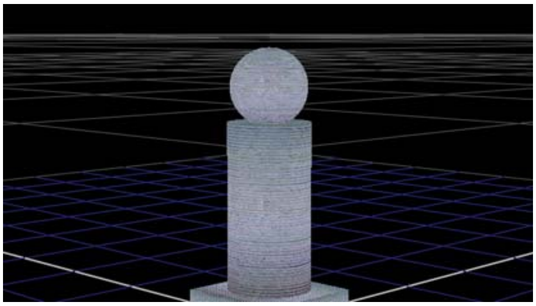


18. Select the **3D** from the view list and rotate the view around the objects in your scene.

    At this point, they have no surface properties, so you'll need to connect a 2D image from the Node Graph to each object.

19. In the Node Graph, connect the **concrete.jpg** to each of the objects.

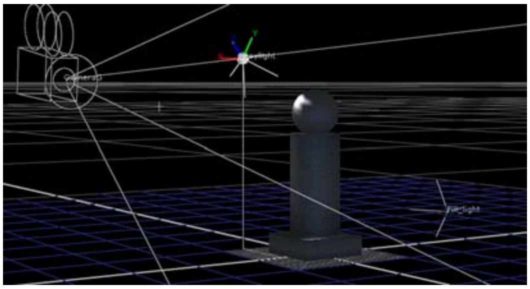




Lighting and Surface Properties

Nuke includes lighting tools to enhance the existing lighting in the plates and images you include in a 3D scene. Also included are fundamental surfacing tools to control the attributes of the objects in the 3D workspace.

These tools are not designed to replace the use of true 3D lighting and surfacing, but they can definitely help you punch up the scene and quickly tweak the settings without sending elements back to the 3D application.

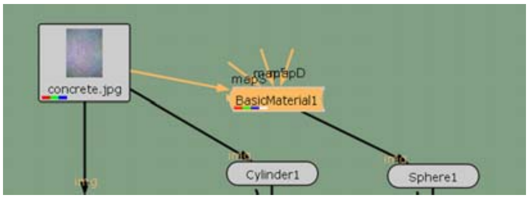


Nuke's lighting objects introduce lighting and surface attributes into your scene. When the scene has no lighting objects, then all surfaces are illuminated with the same properties and level of "brightness."

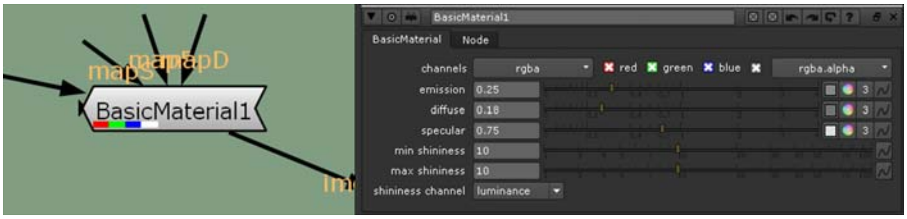
In the following steps, you'll first add nodes that define surface properties for the objects, and then you'll add the light objects to illuminate them.

To define surface attributes to objects:

1. Click on an empty place in the node graph to deselect all nodes. Then, choose **3D > Shader > BasicMaterial**.
2. Drag the **BasicMaterial1** node onto the connector between concrete.jpg and **Sphere1**.



In the Basic Material control panel, you will see parameters to define the amount of light emission, diffusion, and specular properties of the surface. You can mask these properties by connecting images to the **mapS** (specular), **mapE** (emission), and **mapD** (diffuse) connectors, but this is not required for this example.

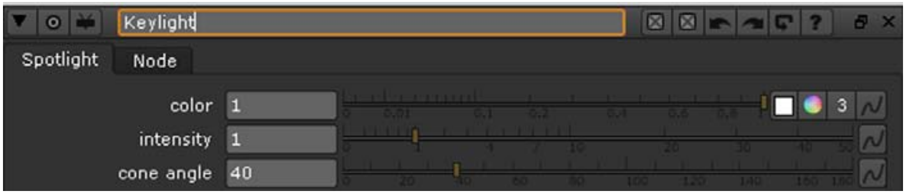


channels rgba ☒ red ☒ green ☒ blue ☒ rgba.alpha
emission 0.25
diffuse 0.18
specular 0.75
min shininess 10
max shininess 10
shininess channel luminance

3. Set the light **emission** control to **0.25**. Set **diffuse** to **0.18**, and **specular** to **0.75**.
4. Adjust the **min shininess** and **max shininess** values to adjust the quality of the specular highlights. Once again, nothing seems to happen! That's because you haven't yet added a light into the scene. All surfaces are illuminated with the same level of brightness, so there are no specular highlights or other controllable light properties visible. It's not very exciting, but don't worry - you get to add a light into the scene soon.
5. Make two copies of the **BasicMaterial1** node and attach a copy before **Cylinder1** and before **Cube1**.

To add light objects to a scene:

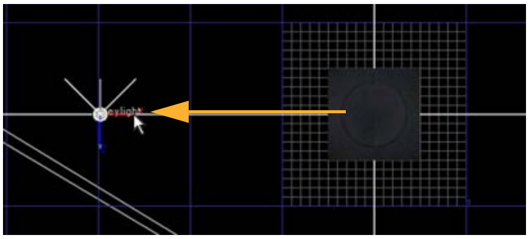
1. Choose **3D > Lights > Spot** and connect the light node to the **Scene2** node.
2. In the **Spotlight1** control panel, rename the light to **Keylight**.



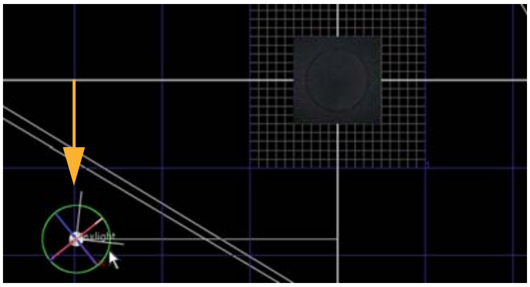
Keylight

color 1
intensity 1
cone angle 40

3. Switch to the **top** view and drag x-axis handle (red) to move the light to the left.

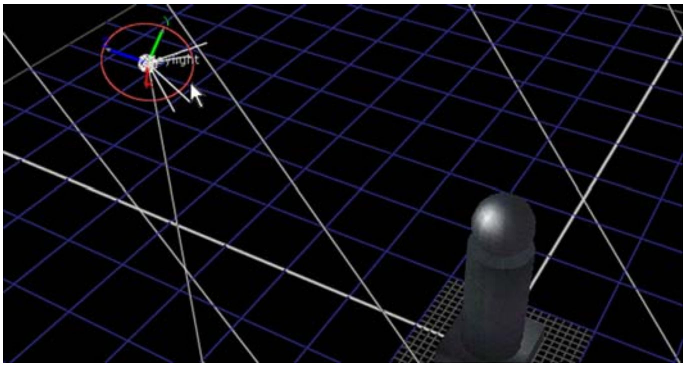


4. Drag the z-axis handle (blue) to move the light closer to the bottom edge of the screen. Then, press the **Ctrl** or **Command** key and rotate the light to face the pillar object.



5. Switch to the **3D** (V) view and rotate the view so you can see both the Keylight and the pillar geometry.
6. Drag the y-axis handle (green) to move the light up above the pillar.

7. Press the **Ctrl** or **Command** key and rotate the light down to shine on the pillar.

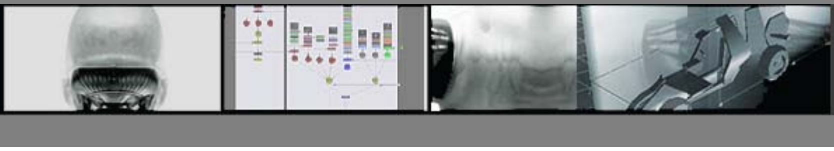


That's the basic setup for lighting and surfaces, but there are other tools for more complex setups. Refer to the Nuke user guide for more information on the 3D lighting and surfacing tools.

Epilog

In this chapter, you learned how to setup a 3D workspace for your composite, and how to work with 3D geometry and lighting. These are all prerequisite skills for more advanced topics, such as camera projections, matchmoving, and set replacement.

- End of Tutorial -



APPENDICES

This section contains supplemental reference information that you may need when using Nuke.

Organisation of the Section

The section consists of four appendices:

- “Appendix A: Hotkeys” lists the keyboard shortcuts you can use for quicker and easier access to Nuke’s features. You can also open a list of keyboard shortcuts from the application by selecting **Help** > **Key Assignments**.
- “Appendix B: Supported File Formats” lists the image and video file formats Nuke supports.
- “Appendix C: Converting from Shake to Nuke” shows you the main differences between Apple’s Shake and Nuke. If you are familiar with Shake but not yet Nuke, we recommend you read this appendix.
- "Appendix D: Third Party Licenses" lists third party libraries used in Nuke, along with their licenses.
- "Appendix E: End User Licensing Agreement" shows you the End User License Agreement that governs the use of Nuke software and this user guide.

APPENDIX A: HOTKEYS

Hotkeys

Keystroke shortcuts, or hotkeys, provide quick access to the features of Nuke. The following tables show these keystrokes.

This appendix assumes you are using the default keyboard and mouse-button assignments. If the mouse buttons do not work for you as described here, try resetting the mouse control type back to the standard Nuke setting (**Edit > Preferences > Viewers > 3D Control Type = Nuke**).

Conventions

The following conventions apply to instructions for mouse-clicks and key presses.

- LMB means click or press the left mouse button.
- MMB means click or press the middle mouse button
- RMB means click or press the right mouse button.
- When you see the word "drag" after a mouse button abbreviation (i.e., **"MMB drag"**), this tells you to press and hold the mouse button while dragging the mouse pointer.
- Keystroke combinations with the **Ctrl**, **Alt**, and **Shift** keys tell you to press and hold the key and then type the specified letter.

  For example, "Press **Ctrl+S**" means hold down the **Ctrl** key, press **S**, and then release both keys.

Important

*On Mac OS X, replace the **Ctrl** key with the **Cmd** key.*

Important

*Keystrokes in the tables appear in upper case, but you do not type them as upper case. If the **Shift+** modifier does not appear before the letter, just press the letter key alone.*

Node Graphs, Viewers, Curve Editors, Script Editors, and Properties Bins

Keystroke(s)	Action
/	Search by node name or class.
Alt+G	Go to a specific frame.
Alt+I	Display script information, such as the node count, channel count, cache usage, and whether the script is in full-res or proxy mode.
Alt+S	Make the active (floating) window fullscreen.
Alt+Shift+S	Save script and increment version number. (See also **Shift+Ctrl+S** under "Viewers".)
Ctrl+F#	Save current window layout. The # represents a function key number, **F1** through **F6**.
Ctrl+I	Open new Viewer window.
Ctrl+LMB on panel name	Float panel.
Ctrl+N	Launch a new project window in a new instance of Nuke.
Ctrl+O	Open a script file.
Ctrl+Q	Exit Nuke.
Ctrl+T	Cycle through tabs in the current pane. Note that this does not work if the focus is on the input pane of the Script Editor.
Ctrl+U	Enable or disable previewing output on an external broadcast video monitor.
Ctrl+Y	Redo last action.
Ctrl+Z	Undo last action.
Shift+Ctrl+S	Save script and specify name (Save As). (See also **Alt+Shift+S** under "Viewers".)
Shift+S	Open Nuke Preferences dialog.
Space bar (short press)	Expand the focused panel to the full window.
Space bar (long press)	Raise the right-click menu.

Properties Panels

Keystroke(s)	Action
up or down arrow	Increment (up) or decrement (down) the value in a parameter field. Click first on the field or press Tab to move focus to the parameter.
Alt+LMB on a close (x) button	Close all properties panels in the Properties Bin.
Alt+LMB drag	Increment (drag left) or decrement (drag right) while dragging over the value in a parameter field.
Alt+R *or* Ctrl+R	Fit Properties Bin to open panels.

Keystroke(s)	Action
Ctrl+A	Select all nodes in the Properties Bin.
Ctrl+Enter (NUM)	Close the current panel.
Ctrl+LMB	Reset slider value to default.
Ctrl+LMB on a close (x) button	Close all properties panels in the Properties Bin except the one clicked on.
Ctrl+LMB drag	Link values between parameter fields.
Ctrl+Return	Close panel (no parameters selected).
Ctrl+Tab *or* Shift+Ctrl+Tab	Move to next tabbed page (**Ctrl+Tab**) or the previous tabbed page (**Shift+Ctrl+Tab**) in the properties panel.
LMB drag	Copy the current value from one parameter field to another.
Return	Chooses selected UI control (default = **OK**).
Shift+Ctrl+A	Close all open properties panels.
Shift+LMB drag	Copy animation (curve or expression) from one parameter field to another.
Tab *or* Shift+Tab	Move focus to next (**Tab**) or previous (**Shift+Tab**) parameter. May need to click on a parameter first, to establish the focus inside the properties panel.

Node Graph

Keystroke(s)	Action
+	Zoom-in (= also zooms-in). See also **LMB+MMB**.
-	Zoom-out.
\	Snaps all nodes to the grid. (See also **Shift+\** under "Node Graph".)
#	Opens a new Viewer window with # representing the number of the connection (**0** to **9**) you want to establish between the new Viewer and the selected node.
.	Inserts Dot node.
up or down arrow	Selects the previous or next node in the tree.
Alt+up arrow	Increment the version number in the selected node's filename.
Alt+down arrow	Decrement the version number in the selected node's filename.
Alt+#	Zoom-out by a specific percentage. The # represents a number between **0** and **9**, with **0**=10%, **1**=100%, **2**=50%, **3**=30%, **4**=25%, **5**=20%, **6**=16%, **7**=14%, **8**=12%, and **9**=11%.
Alt+B	Duplicate and branch selected nodes.
Alt+C	Duplicate selected nodes.
Alt+F	Generate flipbook for selected node using FrameCycler.

Keystroke(s)	Action
Alt+K	Clone selected nodes.
Alt+LMB drag	Pan workspace.
Alt+MMB drag	Zoom-in / zoom-out workspace.
Alt+Shift+K	Remove selected nodes from clone group (declone).
Alt+Shift+U	Splay last selected node to input A.
Alt+U	Splay first selected node to input A.
B	Insert Filter Blur node.
Ctrl	Display connector dots. Drag one to set a dot and create an "elbow."
Ctrl+create node	Replace selected node with the newly created node.
Ctrl+down arrow	Move selected node downstream.
Ctrl+up arrow	Move selected node upstream.
Ctrl+#	Zoom-in by a specific percentage. The # represents a number between **0** and **9**, with **0**=1000%, **1**=100%, **2**=200%, **3**=300%, **4**=400%, **5**=500%, **6**=600%, **7**=700%, **8**=800%, and **9**=900%.
Ctrl+G	Nest selected nodes inside a Group node, replacing the original nodes with the Group.
Ctrl+Alt+LMB on a node	Open the node's properties panel in a floating window.
Ctrl+Alt+G	Replace selected Group node with the nodes nested inside it.
Ctrl+A	Select all nodes in the Node Graph or group window.
Ctrl+B	Node buffer toggle. When this is on, the data upstream from the node is cached or kept in memory, so that it can be read quickly. A yellow line displays under the node to indicate that the caching is on.
Ctrl+C	Copy selected nodes.
Ctrl+D	Disconnect upstream node from selected node.
Ctrl+Alt+Shift+G	Nest selected nodes inside a Group node, keeping the original nodes in the layout. **Ctrl+Enter** opens the new Group node.
Ctrl+LMB on a node	Highlight all upstream nodes.
Ctrl+P	Toggle proxy resolution, as defined on the Settings properties panel. (See also **Ctrl+P** under *Viewers*.)
Ctrl+Return	Open window for selected group node.
Ctrl+Shift+/	Opens the Search and Replace dialog for the selected Read or Write nodes.
Ctrl+Alt+A on a node	Select all nodes in the node tree
Ctrl+Shift+LMB on a node	Select all upstream nodes.
Ctrl+W	Close current script file.
D	Disable / enable selected node.
Delete	Remove selected nodes.

Keystroke(s)	Action
F	Fit the selected nodes (or if no nodes are selected, the entire node tree) to the Node Graph panel or group window.
F12	Clear buffers.
I	Display information for selected node.
K	Insert Copy node. Note that for this to work, you first need to click on the Node Graph to select it. If you have selected the Viewer, pressing K stops playback. (See **K** under "Viewers".)
LMB+MMB	Drag to zoom in the Node Graph.
M	Insert Merge node.
MMB	Hold and drag to pan in the Node Graph. Click to fit selected nodes (or entire tree) to screen.
N	Rename the selected node.
O	Insert Roto node.
P	Insert RotoPaint node.
Return	Open panel for selected node(s).
Shift+\	Snaps selected node to the grid. (See also \ under "Node Graph".)
Shift+0, 1, 2, 3, ...	Connect the selected node to Viewer as reference input.
Shift+create node	Create a node in a new branch of the node tree.
Shift+drag	Duplicate selected arrow.
Shift+A	Insert an AddMix node.
Shift+Ctrl+C	Set color for selected nodes.
Shift+Ctrl+X	Extract selected nodes.
Shift+U	Splay selected nodes to last selected node.
Shift+X	Swap A/B inputs on selected node.
Shift+Y	Connect second selected node to the output of the first selected node.
Tab	Open a text field where you can enter the first letters of a node name to bring up a list of matches. Press **Return** to insert a node from the list.
U	Splay selected nodes to first selected node.
Y	Connect first selected node to the output of the second selected node.

Editing

Keystroke(s)	Action
Alt+Ctrl+V	Paste knob values.
Backspace	Erase or Delete Left.

Keystroke(s)	Action
Ctrl+B	Left justify selected text.
Ctrl+C	Copy.
Ctrl+E	Move cursor to end of selected text.
Ctrl+F	Right justify selected text.
Ctrl+K	Delete text from the cursor to the next space.
Ctrl+N	Bottom justify selected text.
Ctrl+P	Top justify selected text.
Ctrl+V	Paste.
Ctrl+X	Cut.
Shift+Ctrl+V	Paste (see **Ctrl+V**).

Viewers

Keystroke(s)	Action
-	Zoom-out. (See also **LMB+MMB**.)
+	Zoom-in (= also zooms-in).
.	Gain display, increase.
; (semicolon)	Switch to the previous view in a multiview project.
' (forward single quote)	Switch to the next view in a multiview project.
!	Turn on Viewer "blend," split-screen display. (See also **W** under *Viewers* for toggle on/off).
{	Show / hide top toolbar.
}	Show / hide bottom toolbar.
0, 1, 2, 3, ...	Establish a numbered connection (**1** - **9**, **0**) between the selected node and the active Viewer. Displays the node's output in that Viewer.
Shift+0, 1, 2, 3, ...	Connect reference inputs to Viewer.
Numeric keypad	Nudge on-screen controls left, right, up, or down. (See also **Shift+numeric keypad** under "Viewers".)
Right arrow	Step forward one frame.
Left arrow	Step back one frame.
, (comma)	Gain display, decrease.
` (accent key)	Show / hide all Viewers.
A	Display the alpha channel or the channel displayed in the list at the top of the Viewer.
Alt+left arrow	Previous keyframe.

Keystroke(s)	Action
Alt+right arrow	Next keyframe.
Alt+*#*	Zoom-out by a specific percentage. The # represents a number between **0** and **9**, with **0**=10%, **1**=100%, **2**=50%, **3**=30%, **4**=25%, **5**=20%, **6**=16%, **7**=14%, **8**=12%, and **9**=11%.
Alt+G	Go to specific frame.
Alt+LMB drag	Pan inside the Viewer window.
Alt+MMB drag	Zoom in (drag right) or out (drag left) in the Viewer window.
Alt+P	Open the controls of the currently active Viewer process.
Alt+R	Resize Viewer to image (see also **Ctrl+R** under *Viewers*).
Alt+Shift+R	Resize Viewer and image to fit frame.
Alt+W	Activate the ROI feature, or if a ROI exists, clear the current ROI. Drag to define a new ROI. (See also **Shift+W** under "Viewers".)
Alt+Z	Toggle lock/unlock the Viewer to a specified zoom level for all inputs.
B	Display blue channel / RGB toggle.
Backspace	Cycle through Viewer inputs in reverse order. If wipe is active, cycles through inputs on the left-hand side.
Ctrl+right arrow	Move to midpoint between current frame and next keyframe/last frame.
Ctrl+left arrow	Move to midpoint between current frame and previous keyframe/first frame.
Ctrl+#	Zoom-in by a specific percentage. The # represents a number between **0** and **9**, with **0**=1000%, **1**=100%, **2**=200%, **3**=300%, **4**=400%, **5**=500%, **6**=600%, **7**=700%, **8**=800%, and **9**=900%.
Ctrl+Alt+LMB	Sample a single pixel's color value from the node's input while viewing its output. (See also **Ctrl+LMB** under "Viewers".)
Ctrl+Alt+Shift+LMB	Sample range of pixels from the node's input while viewing its output. (See also **Ctrl+Shift+LMB** under "Viewers".)
Ctrl+LMB	Sample a single pixel's color value from the Viewer. (See also **Ctrl+Alt+LMB** under "Viewers".)
Ctrl+P	With the mouse pointer over the Viewer, this keystroke toggles pixel aspect ratio between square and non-square, according to the setting of the default format under the Settings properties panel. This is not the same as toggling proxy resolution (see also **Ctrl+P** under *Node Graph*).
Ctrl+R	Resize Viewer window to image (see **Alt+R** under *Viewers*).
Ctrl+Shift+LMB	Sample range of pixels from the Viewer. (See also **Ctrl+Alt+Shift+LMB** under "Viewers".)
Ctrl+U	Enable or disable previewing output on an external broadcast video monitor.
End	Go to last frame.
Esc	Close Viewer.
F	Fit image to Viewer.
G	Display green channel / RGB toggle.
H	Fill image in Viewer.

Keystroke(s)	Action
J	Play backward.
K	Stop playback. Note that for this to work, you first need to click on the Viewer to select it. If you have selected the Node Graph, pressing K inserts a Copy node. (See **K** under "Node Graph".)
L	Play forward.
LMB+MMB	Drag to zoom in the Viewer.
M	Display Matte, or alpha channel as transparent overlay.
MMB	Drag to pan inside the Viewer window. Click to fit Viewer to frame.
O	Show / hide overlays.
P	Disable (pause) the display refresh of the Viewer.
R	Display red channel / RGB toggle.
RMB (or press and hold the spacebar)	Display Viewer menu.
S	Display Viewer Settings dialog.
Shift+left arrow	Move left on the timeline by the specified increment amount.
Shift+right arrow	Move right on the timeline by the specified increment amount.
Shift+A	Display "other" channel / RGB toggle. Current input only. (Default = Alpha).
Shift+B	Display Blue channel / RGB toggle. Current input only.
Shift+Backspace	Activate Wipe. Cycle images on right side.
Shift+Ctrl+R	Resize Viewer to maximum and fit image.
Shift+F	Maximum Viewer window toggle.
Shift+G	Display Green channel / RGB toggle. Current input only.
Shift+L	Display Luminance / RGB toggle. Current input only.
Shift+M	Display Matte / RGB toggle. Current input only.
Shift+numeric keypad	Nudge on-screen controls left, right, up, or down by increment. (See also **Numeric keypad** under "Viewers".)
Shift+R	Display Red channel / RGB toggle. Current input only.
Shift+W	Region of interest (ROI) toggle. (See also **Alt+W** under Viewers.)
Tab	2D / 3D view toggle.
U	Update Viewer display. Used when Pause is active (press **P**).
W	Toggles Viewer "blend," split-screen display, on/off. This was previously called the "wipe" mode. (See also ! under *Viewers*).

3D Viewer

Keystroke(s)	Action
Alt+LMB	Translate Viewer perspective on y (drag up/down) or z (drag left/right).
Alt+MMB	Zoom Viewer perspective in (drag right) or out (drag left).
Alt+RMB	Rotate Viewer perspective on x (drag up/down) or y (drag left/right).
Ctrl+LMB	Rotate Viewer perspective on x (drag up/down) or y (drag left/right). Ctrl+LMB on the 3D view mode button activates Interactive mode.
Ctrl+L	Toggle between Unlocked and Locked 3D view mode for selected Camera or Light. (See Ctrl+LMB if you want to activate the Interactive mode.)
Ctrl+Shift+LMB	Rotate Viewer perspective on z.
Shift+C	3D view, bottom orthographic.
Shift+X	3D view, left-side orthographic.
Shift+Z	3D view, back orthographic.
Tab	3D / 2D view toggle.
V	3D view, perspective.
X	3D view, right-side orthographic.
Z	3D view, front-side orthographic.

RotoPaint Draw

Keystroke(s)	Action
C	Toggle between Clone and Reveal tools.
Ctrl+A	Select all points.
Ctrl+Alt+click (when editing a stroke/shape)	Add a point to a stroke/shape.
Ctrl+click (when editing points)	Break tangent handles for selected points.
Backspace (on the stroke/shape list)	Delete an item from the stroke/shape list. Note that if the RotoPaint node is selected in the Node Graph, pressing Backspace deletes the entire node.
Ctrl+click (when drawing a Bezier or B-spline shape)	Sketch a Bezier or a B-spline shape.
Ctrl+drag (when the Clone or Reveal Tool is active)	Set the offset between the source and destination.

Keystroke(s)	Action
Ctrl+Shift+drag (when the B-spline Tool is active)	Increase (drag right) and decrease (drag left) tension of the B-spline shape.
D	Toggle between Dodge and Burn tools.
Delete	Remove selected points.
E	Increase feather for selected points.
I	Pick color.
N	Toggle between Brush and Eraser tools.
Q	Toggle between Select All, Select Curves and Select Points tools.
Return (when the Bezier or B-spline tool is active)	Close shape.
Shift+click (when drawing a Bezier shape)	Create a sharp point on the previous point.
Shift+click (when editing points in a stroke/shape)	Bring up a transform box for the points selected.
Shift+drag (when editing points in a Bezier or B-spline shape)	Move both tangent handles at the same time.
Shift+drag (when the Brush, Eraser, Clone or Reveal tool is active)	Change brush size.
Shift+E	Remove feather outline from selected points.
Shift+Z	Make selected points linear (Cusp).
T (when Select tool active)	Display a transform box (for points) or a transform jack (for a whole shapes).
T (when Clone tool active)	Show/hide source as onion skin with transform box/jack.
Z	Make tangent handles horizontal on selected points (Smooth).
V	Toggle between Bezier, B-Spline, Ellipse and Rectangle tools.
X	Toggle between Blur, Sharpen, and Smear tools.

Curve Editor and Dope Sheet

Keystroke(s)	Action
Alt+LMB drag	Pan inside the Curve Editor.
Alt+Shift+LMB drag	Move a single point, leaving any other selected points where they are.
Alt+MMB drag	Variable zoom: zoom in or out on the x or y axis only.
C	Change interpolation of selected control points to Cubic.
Ctrl+A	Select all curves.
Ctrl+Alt+LMB	Add a point to the current curve. In the Dope Sheet, add a keyframe.
Ctrl+Alt+Shift+LMB drag	Sketch points freely on the current curve.
Ctrl+C	Copy selected keys.
Ctrl+E	Copy expressions.
Ctrl+L	Copy links.
Ctrl+LMB drag	Remove horizontal/vertical constraint on moving points.
Hold down Ctrl+Shift	Hide points to make it easier to click on the selection box or transform jack.
Ctrl+V	Paste curve.
Ctrl+X	Cut selected keys.
H	Change interpolation of selected control points to Horizontal.
K	Change interpolation of selected control points to Constant.
L	Change interpolation of selected control points to Linear.
LMB	Select a single point on the curve.
LMB+MMB	Drag to zoom in the Curve Editor.
LMB drag on blank space	Draw a box to select multiple points.
LMB drag on a point	Move all selected points.
LMB drag on a selection box	Resize a selection box and scale the points inside.
LMB drag on a transform jack	Move all points inside the selection box.
MMB or F	Fit selection in the window.
MMB drag	Draw a box around an area and zoom to fit that area in the Editor.
R	Change interpolation of selected control points to Catmull-Rom.
Shift+Ctrl+C	Copy selected curves.
Shift+LMB	Add or remove points to/from selection.
Shift+LMB drag	Draw box to add/remove points to/from selection.

Keystroke(s)	Action
X	Break selected control points' handles.
Z	Change interpolation of selected control points to Smooth (Bezier).

Script Editor

Keystroke(s)	Action
Ctrl+[	Step back to the previous statement.
Ctrl+]	Step forward to the next statement.
Ctrl+Backspace	Clear output pane.
Ctrl+Enter (numeric keypad)	Run the script in the Editor.
Ctrl+Return	Run the script in the Editor.
Ctrl+Shift+[	Decrease the indentation level of selected text.
Ctrl+Shift+]	Increase the indentation level of selected text.
Shift+Tab	Decrease the indentation level.
Tab	Increase the indentation level.

Toolbar

Keystroke(s)	Action
MMB	Repeat the last item used from the menu.

Content Menus

Keystroke(s)	Action
Ctrl+LMB	Open the selected menu item in a floating window.

Color Picker

Keystroke(s)	Action
Drag right or left (on a slider label)	Scrub the value up or down.
Alt+drag right or left (on a slider label)	Scrub the value up or down slowly.
Alt+LMB (on a slider label)	Decrement the value by 0.001.
Alt+RMB (on a slider label)	Increment the value by 0.001.
LMB (on a slider label)	Decrement the value by 0.01.
RMB (on a slider label)	Increment the value by 0.01.
Shift+drag right or left (on a slider label)	Scrub the value up or down fast.
Shift+LMB (on a slider label)	Decrement the value by 0.1.
Shift+RMB (on a slider label)	Increment the value by 0.1.

APPENDIX B: SUPPORTED FILE FORMATS

Supported File Formats

This appendix lists the image and video formats recognized by Nuke. When importing and exporting image sequences remember the following:

- When you import images with a Read node (Image > Read), Nuke analyzes the contents of the file to determine the format.The filename extension is not used to determine file format, which allows flexibility with naming conventions in a production environment.
- Regardless of format, Nuke converts all imported sequences to its native 32-bit linear RGB colorspace.
- When you render new images from Nuke (Image > Write), you can use a filename extension to specify format.

Supported Image Formats

The following table lists the supported image formats. The extensions listed under "Filename Extension" let you specify the image format; use these as the actual filename extensions or the prefix to indicate output format for the image sequences.

Format	Bit Depths	Read/Write	Extension	Notes
AVI	n/a	read and write	avi	On Windows, in order to support more codecs, the AVI reader uses the DirectShow multimedia architecture. When decoding AVI files, DirectShow tries to find the appropriate codec on the system. If the codec is not available, DirectShow and Nuke are unable to open the AVI file. Note that the 64-bit version of Nuke can only use 64-bit DirectShow codecs. If you only have a 32-bit codec installed, the 64-bit version of Nuke cannot use it to open AVI files.
CIN	10 (log)	read and write	cin	
DPX	8, 10, 12, and 16	read and write	dpx	
EXR	16 and 32	read and write	exr	Exr handles 16- and 32-bit float. This 16 is also called "half float" and is different from the 16-bit integer that all the other formats that support 16 use.
FPI	obsolete			
GIF	8	read only	gif	
Radiance	16	read and write	hdr, hdri	This format stores an 8-bit mantissa for each of r, g, and b and an additional 8-bit exponent that is shared by all three, which packs the floating point RGB triplet into 32 bits per pixel.

Format	Bit Depths	Read/Write	Extension	Notes
JPEG	8	read and write	jpg, jpeg	Adjust compression levels using the quality slider in the Write node's properties panel.
Maya IFF	8 and 16	read only	iff	
PNG	8 and 16	read and write	png (8-bit) png16 (16-bit)	
PSD	8	read only	psd	Nuke doesn't read layer comps or adjustment layers, or recognize layer or group blend modes. Layers are read into separate Nuke channel sets and anything that doesn't map into that (for example, a blend mode) is ignored.
QuickTime	n/a	read and write	mov	QuickTime is only supported by default on 32-bit Windows and Mac OS X (32-bit and 64-bit). To use QuickTime files on Linux, you need to use the prefix ffmpeg: before the file path and file name, for example, ffmpeg:\z:\job\FILM\IMG\final_comp_v01.####.mov. When working with Write nodes, you can also choose ffmpeg from the file type menu and use mov as the file extension. This way, Nuke will use its reader/writer that is based on the FFmpeg open source library to decode/encode QuickTime files.
RAW	n/a	read only	n/a	DSLR raw data files, such as Canon .CR2 files. These are only supported via the dcraw command-line program, which you can download from the dcraw website. Bit depth and other specifications depend on the device. Some devices may not be supported.
REDCODE	16	read only	r3d	Note that .r3d files may look different in Nuke compared to various versions of RED applications, like RED ALERT or REDCINE. Unlike most other file formats Nuke reads, the .r3d REDCODE files must be processed to convert from a raw format to an RGB color image. From time to time, a new version of the RED SDK that Nuke uses improves this processing and due to the timing of release cycles, Nuke may sometimes be using a different version than the RED applications.
SGI	8 and 16	read and write	sgi, rgb, rgba (8-bit sequences) sgi16 (for 16-bit sequences)	
SoftImageâ PIC	8	read and write	pic	

Format	Bit Depths	Read/Write	Extension	Notes
TIFF	8, 16, and 32	read and write	tif, tiff (8-bit sequences) tif16, tiff16 (16-bit sequences) ftif, ftiff (32-bit sequences)	If utilized, the compression schema on imported TIFF sequences must be LZWâ.
Truevisionâ TARGA	8	read and write	tga, targa	
Wavefrontâ RLA	8	read only	rla	
XPM	8	read and write	xpm	This is the text-based format in which Nuke's interface elements are stored.
YUV	8	read and write	yuv	This format does not specify resolution, so Nuke assumes a width of 720 pixels.

APPENDIX C: CONVERTING FROM SHAKE TO NUKE

Converting from Shake to Nuke

This appendix contains information that will assist Shake users in making the transition to Nuke. Although Nuke does provide an intuitive workflow for a broad base of compositing tasks, there will be a few cases where Shake artists may ask "How do I (*fill in the blank)* in Nuke?" The following information will help you fast-track to the answers you need.

The first part of this appendix provides practical information for Shake artists, including terms used in Nuke, the layout of the user interface, and differences in the workflow for common tasks. The second part lists the commands you use to create node trees in Shake with their counterparts in Nuke.

Terms (and Conditions)

Let's start with vocabulary differences between Shake and Nuke. The following table lists several Shake terms and their Nuke equivalents. This table does not include the names of operators you use to create a node tree; these are covered later in this appendix.

Shake Term	Nuke Equivalent	Description	More Info
Cache	Cache or buffer	Cached or buffered image data is saved to disk so Nuke can quickly display the results without recalculating the parts of the script that have not changed.	Nuke User Guide, page 129
Command Palette	Toolbar	The palette that contains all possible nodes for creating your node tree. Shake's command palette lets you browse through the nodes according to category. In Nuke, you can do the same by rolling over the buttons on the Toolbar.	Nuke User Guide, page 44
Console	(see description)	To display commands as they are executed in Nuke, you must launch Nuke from a shell using the -V option. For example, Windows users would open a command line window, navigate to Nuke application directory and enter: `nuke6.2.exe -V`	
Control	Knob or control	A control that appears in the parameters of a node, such as a check box that toggles an option or a slider to change a numeric value. You can create custom "knobs" to add new controls to the Nuke properties panels.	Nuke User Guide, page 60
Globals	Project Settings	The place where you define the global settings for the current script, such as resolution, frame range, frames per second, and other project settings.	Nuke User Guide, page 122

Shake Term	Nuke Equivalent	Description	More Info
Interface Setting	Window Layout	Nuke's interface is highly customizable. When you find an arrangement of panes and panels that you like, your can save the window layout. Press **Ctrl/Cmd+F1** to save the layout. Press **Shift+F1** to restore. You can save up to 6 layouts (Ctrl/Cmd+F1, Ctrl/Cmd+F2, Ctrl/Cmd+F3...). To restore a layout, just press **Shift** and the function key you used to save the layout.	Nuke User Guide, page 116
Layer	Merge	The process of compositing one image with another. You know it as layering in Shake. In Nuke, it's called Merge.	
Macro	Gizmo	These are the user-defined nodes, where a series of operations are saved, tweaked, and presented as group or object that may be used again in other scripts. When defining a gizmo, you decide which "knobs" (controls) are displayed from the original group of nodes.	Nuke User Guide, page 629
Node View	Node Graph or DAG	The main workspace where you construct your node tree. In Nuke, this is sometimes referred to as "The DAG," which is old-school lingo for "The Directed Acyclic Graph."	Nuke User Guide, page 40
Noodle	Connector or pipe	The lines that connect the items in the node tree. "Shaking" a node does not disconnect it from its tree in Nuke, but pressing **Ctrl/Cmd+Shift+X** will disconnect (or extract) the node.	Nuke User Guide, page 52
Parameters	Properties panel or control panel	These are the controls that determine what a node passes to the next node in the tree. In Nuke, you can open an unlimited number of properties panels to edit the nodes in your tree—much better than Shake's limit of two parameter tabs.	Nuke User Guide, page 60
Thumbnail	Postage Stamp	The preview image on a node. In Nuke, you can show or hide a thumbnail for any node. They are refreshed automatically as changes are made to your script.	

Node Reference

These tables list the nodes as organized in the Shake user interface. Where applicable, the Nuke equivalent is listed with notes that describe any differences and conditions that may be useful to the compositing artist.

Image Nodes

Shake Node	Nuke Equivalent	Description	More Info
Checker	Image > Checker-board		
Color	Image > Constant		
FileIn	Image > Read		Nuke User Guide, page 134
FileOut	Image > Write		Nuke User Guide, page 523
QuickPaint	Draw > RotoPaint		Nuke User Guide, page 329
QuickShape	Draw > Roto		Nuke User Guide, page 329
Ramp	Draw > Ramp		
RGrad	Draw > Radial		
RotoShape	Draw > RotoPaint		Nuke User Guide, page 329
Text	Draw > Text		

Color Nodes

Shake Node	Nuke Equivalent	Description	More Info
Add	Color > Math > Add		Nuke User Guide, page 221
AdjustHSV	Color > HSVTool		Nuke User Guide, page 209
Brightness	Color > Multiply		Nuke User Guide, page 222
Clamp	Color > Math > Clamp		Nuke User Guide, page 220

Shake Node	Nuke Equivalent	Description	More Info
ColorCorrect	Color > ColorCorrect		Nuke User Guide, page 206
ColorSpace	Color > Colorspace		Nuke User Guide, page 224
ColorX	Color > Math > Expression		Nuke User Guide, page 222
ContrastRGB	Color > RolloffContrast		
Fade	Color > Multiply		Nuke User Guide, page 222
Gamma	Color > Math > Gamma		
HueCurves	Color > HueCorrect		Nuke User Guide, page 212
Invert	Color > Math > Invert		
LogLin	Color > Log2Lin		Nuke User Guide, page 224
Lookup	Color > ColorLookup		Nuke User Guide, page 206
LookupFile	Color > ColorLookup		Nuke User Guide, page 206
MDiv	Merge > Unpremult		
MMult	Merge > Premult		
Reorder	Channel > Shuffle		Nuke User Guide, page 180
Saturation	Color > Saturation		Nuke User Guide, page 214
Truelight	Color > Truelight		

Filter Nodes

Shake Node	Nuke Equivalent	Description	More Info
Blur	Filter > Blur		
Convolve	Filter > Convolve		
Defocus	Filter > Defocus		
DilateErode	Filter > Erode		

Shake Node	Nuke Equivalent	Description	More Info
EdgeDetect	Filter > EdgeDetect		
Emboss	Filter > Emboss		
FilmGrain	Draw > ScannedGrain		Nuke User Guide, page 217
Median	Filter > Median		
PercentBlur	Filter > Blur		
RBlur	Filter > Godrays		
Sharpen	Filter > Sharpen		
Zblur	Filter > ZBlur		

Key Nodes

Shake Node	Nuke Equivalent	Description	More Info
ChromaKey	Keyer > HueKeyer		
DepthSlice	Filter > ZSlice		
LumaKey	Keyer > Keyer		
Primatte	Keyer > Primatte		Nuke User Guide, page 261
Keylight	Keyer > Keylight		Nuke User Guide, page 298

Layer Nodes

Shake Node	Nuke Equivalent	Description	More Info
AddMix	Merge > AddMix		
AddText	Draw > Text	May be added "in-line" to composite text over an input without the use of a layer (merge) node. In Shake, this was added because the Image > Text node did not have an input. In Nuke, the Draw > Text node may be used in-line or layered over another image with a Merge operation.	
Atop	Merge > Merge	Set operator = atop	
Copy	Channel > Copy		Nuke User Guide, page 182
IAdd	Merge > Merge	Set operator = plus	Nuke User Guide, page 185
IDiv	Merge > Merge	Set operator = divide	Nuke User Guide, page 185
IMult	Merge > Merge	Set operator = multiply	Nuke User Guide, page 185
Inside	Merge > Merge	Set operator = in	Nuke User Guide, page 185
ISub	Merge > Merge	Set operator = minus	Nuke User Guide, page 185
ISubA	Merge > Merge	Set operator = difference	Nuke User Guide, page 185
KeyMix	Merge > Keymix		
Max	Merge > Merge	Set operator = max	Nuke User Guide, page 185

Shake Node	Nuke Equivalent	Description	More Info
Min	Merge > Merge	Set operator = min	Nuke User Guide, page 185
Mix	Merge > Dissolve		
Outside	Merge > Merge	Set operator = out	Nuke User Guide, page 185
Over	Merge > Merge	Set operator = over	Nuke User Guide, page 185
Screen	Merge > Merge	Set operator = screen	Nuke User Guide, page 185
Under	Merge > Merge	Set operator = under	Nuke User Guide, page 185
Xor	Merge > Merge	Set operator = xor	Nuke User Guide, page 185
ZCompose	Merge > ZMerge		

Other Nodes

Shake Node	Nuke Equivalent	Description	More Info
PixelAnalyzer	Image > CurveTool		Nuke User Guide, page 422

APPENDIX D: THIRD PARTY LICENSES

Third Party Licenses

This appendix lists third party libraries used in Nuke, along with their licenses.

Library	Description	License
Boost	Source code function / template library	Boost Software License - Version 1.0 - August 17th, 2003 Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following: The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
Exif	Metadata parser	author Lutz Mueller lutz@users.sourceforge.net date 2001-2005 This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

Library	Description	License
Expat	XML parser	Copyright © 1998, 1999, 2000 Thai Open Source Software Center Ltd and Clark Cooper Copyright © 2001, 2002, 2003, 2004, 2005, 2006 Expat maintainers. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
Autodesk FBX	File format support	This software contains Autodesk® FBX® code developed by Autodesk, Inc. Copyright 2008 Autodesk, Inc. All rights, reserved. Such code is provided "as is" and Autodesk, Inc. disclaims any and all warranties, whether express or implied, including without limitation the implied warranties of merchantability, fitness for a particular purpose or non-infringement of third party rights. In no event shall Autodesk, Inc. be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of such code.
FreeType	Font support	Portions of this software are copyright © 2008 The FreeType Project (www.freetype.org). All rights reserved.

Library	Description	License
GLEW	OpenGL support	The OpenGL Extension Wrangler Library Copyright (C) 2002-2008, Milan Ikits <milan ikits□ieee org> Copyright (C) 2002-2008, Marcelo E. Magallon <mmagallo□debian org> Copyright (C) 2002, Lev Povalahev All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * The name of the author may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
FTGL	OpenGL support	FTGL - OpenGL font library Copyright © 2001-2004 Henry Maddocks ftgl@opengl.geek.nz Copyright © 2008 Sam Hocevar sam@zoy.org Copyright © 2008 Sean Morrison learner@brlcad.org Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
IJG JPEG	File format support	This software is based in part on the work of the Independent JPEG Group.

Library	Description	License
Lib Tiff	File Format Support	Copyright © 1988-1997 Sam Leffler Copyright © 1991-1997 Silicon Graphics, Inc. Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics. THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
OFX	Plug-in API	Copyright (c) 2003-2009, The Open Effects Association Ltd. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: • Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. • Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. • Neither the name The Open Effects Association Ltd, nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Library	Description	License
OpenEXR	File format support	Copyright © 2002, Industrial Light & Magic, a division of Lucas Digital Ltd. LLC All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * Neither the name of Industrial Light & Magic nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
Poisson Surface Reconstruction	Source code library	Copyright (c) 2006, Michael Kazhdan and Matthew Bolitho All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of the Johns Hopkins University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
Python	Source code language	Copyright © 2001, 2002, 2003, 2004 Python Software Foundation; All Rights Reserved.

Library	Description	License
TCL	Source code language	This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., Scriptics Corporation, and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files. The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply. IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS. GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 © (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7013 © (1) of DFARs. Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

Library	Description	License
VXL	Computer vision	Copyright © 2000-2003 TargetJr Consortium GE Corporate Research and Development (GE CRD) 1 Research Circle Niskayuna, NY 12309 All Rights Reserved Reproduction rights limited as described below. Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notice and this permission notice appear in all copies of the software and related documentation, (ii) the name TargetJr Consortium (represented by GE CRD), may not be used in any advertising or publicity relating to the software without the specific, prior written permission of GE CRD, and (iii) any modifications are clearly marked and summarized in a change history log. THE SOFTWARE IS PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL THE TARGETJR CONSORTIUM BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR ON ANY THEORY OF LIABILITY ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
SparseLib++	Source code library	SparseLib++ : Sparse Matrix Library National Institute of Standards and Technology University of Notre Dame Authors: R. Pozo, K. Remington, A. Lumsdaine NOTICE Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted provided that the above notice appear in all copies and supporting documentation. Neither the Institutions (National Institute of Standards and Technology, University of Notre Dame) nor the Authors make any representations about the suitability of this software for any purpose. This software is provided "as is" without expressed or implied warranty.
InventorMath	Maths library	Open Inventor code is copyright SGI. All rights reserved.

APPENDIX E: END USER LICENSING AGREEMENT

End User Licensing Agreement (EULA)

This END USER SOFTWARE LICENSE AGREEMENT (this "Agreement") is made by and between The Foundry Visionmongers Ltd., a company registered in England and Wales, ("The Foundry"), and you, as either an individual or a single entity ("Licensee").

In consideration of the mutual covenants contained herein and for other good and valuable consideration (the receipt and sufficiency of which is acknowledged by each party hereto) the parties agree as follows:

SECTION 1. GRANT OF LICENSE.

Subject to the limitations of Section 2, The Foundry hereby grants to Licensee a limited, non-transferable and non-exclusive license to install and use a machine readable, object code version of this software program (the "Software") and the accompanying user guide and other documentation (collectively, the "Documentation") solely for Licensee's own internal business purposes (collectively, the "License"); provided, however, Licensee's right to install and use the Software and the Documentation is limited to those rights expressly set out in this Agreement.

SECTION 2. RESTRICTIONS ON USE.

Licensee is authorized to use the Software in machine readable, object code form only, and Licensee shall not: (a) assign, sublicense, transfer, pledge, lease, rent, share or export the Software, the Documentation or Licensee's rights hereunder; (b) alter or circumvent the copy protection mechanisms in the Software or reverse engineer, decompile, disassemble or otherwise attempt to discover the source code of the Software; (c) modify, adapt, translate or create derivative works based on the Software or Documentation; (d) use, or allow the use of, the Software or Documentation on any project other than a project produced by Licensee (an "Authorized Project"); (e) allow or permit anyone (other than Licensee and Licensee's authorized employees to the extent they are working on an Authorized Project) to use or have access to the Software or Documentation; (f) copy or install the Software or Documentation other than as expressly provided for herein; or (g) take any action, or fail to take action, that could adversely affect the trademarks, service marks, patents, trade secrets, copyrights or other intellectual property rights of The Foundry or any third party with intellectual property rights in the Software (each, a "Third Party Licensor"). Furthermore, for purposes of this Section 2, the term "Software" shall include any derivatives of the Software.

Licensee shall install and use only a single copy of the Software on one computer, unless the Software is installed in a "floating license" environment, in which case Licensee may install the Software on more than one computer; provided, however, Licensee shall not at any one time use more copies of the Software than the total number of valid Software licenses purchased by Licensee.

Furthermore, the Software can be licensed on an "interactive" or "non-interactive" basis. Licensee shall be authorized to use a non-interactive version of the Software for rendering purposes only (i.e., on a CPU, without a user, in a non-interactive capacity) and shall not use such Software on workstations or otherwise in a user-interactive capacity. Licensee shall be authorized to use an interactive version of the Software for both interactive and non-interactive rendering purposes.

Finally, if the Software is a "Personal Learning Edition", Licensee may use it only for the purpose of training and instruction, and for no other purpose. PLE versions of the Software may not be used for commercial, professional or for-profit purposes.

SECTION 3. BACK-UP COPY.

Notwithstanding Section 2, Licensee may store one copy of the Software and Documentation off-line and off-site in a secured location owned or leased by Licensee in order to provide a back-up in the event of destruction by fire, flood, acts of war, acts of nature, vandalism or other incident. In no event may Licensee use the back-up copy of the Software or Documentation to circumvent the usage or other limitations set forth in this Agreement.

SECTION 4. OWNERSHIP.

Licensee acknowledges that the Software and Documentation and all intellectual property rights relating thereto are and shall remain the sole property of The Foundry and the Third Party Licensors. Licensee shall not remove, or allow the removal of, any copyright or other proprietary rights notice included in and on the Software or Documentation or take any other action that could adversely affect the property rights of The Foundry or any Third Party Licensor. To the extent that Licensee is authorized to make copies of the Software or Documentation under this Agreement, Licensee shall reproduce in and on all such copies any copyright and/or other proprietary rights notices provided in and on the materials supplied by The Foundry hereunder. Nothing in this Agreement shall be deemed to give Licensee any rights in the trademarks, service marks, patents, trade secrets, copyrights or other intellectual property rights of The Foundry or any Third Party Licensor, and Licensee shall be strictly prohibited from using the name, trademarks or service marks of The Foundry or any Third Party Licensor in Licensee's promotion or publicity without The Foundry's express written approval.

SECTION 5. LICENSE FEE.

Licensee understands that the benefits granted to Licensee hereunder are contingent upon Licensee's payment in full of the license fee payable in connection herewith (the “License Fee”).

SECTION 6. UPGRADES/ENHANCEMENTS.

As long as Licensee pays the applicable fee for annual support, upgrades and updates (“Annual Upgrade and Support Program”), The Foundry or its authorized reseller will provide Licensee with access to upgrades and updates, if any, made available by The Foundry on the terms and conditions set forth in this Agreement, unless such upgrade or update contains a separate license. The term shall begin upon the date of purchase and continue for 12 months, unless otherwise expressly agreed to in writing by The Foundry. Before requesting technical support, purchasers of the Annual Upgrade and Support Program must send an email to support@thefoundry.co.uk in order to register the names of up to 2 employees who shall then become the technical representatives of the purchaser. The Foundry will endeavour to provide news of upgrades and technical support to the technical representatives; however, The Foundry cannot guarantee resolution or the results of any assistance that may be provided. The Foundry's technical support personnel can be contacted by phone Monday through Friday (excluding holidays) during normal business hours 0930-1800, and by email at support@thefoundry.co.uk. Subsequent renewals of the Annual Upgrade and Support Program will be charged at the rate shown in The Foundry's current price list.

SECTION 7. TAXES AND DUTIES.

Licensee agrees to pay, and indemnify The Foundry from claims for, any local, state or national tax (exclusive of taxes based on net income), duty, tariff or other impost related to or arising from the transaction contemplated by this Agreement.

SECTION 8. LIMITED WARRANTY.

The Foundry warrants that, for a period of ninety (90) days after delivery of the Software: (a) the machine readable electronic files constituting the Software and Documentation shall be free from errors that may arise from the electronic file transfer from The Foundry and/or its authorized reseller to Licensee; and (b) to the best of The Foundry's knowledge, Licensee's use of the Software in accordance with the Documentation will not, in and of itself, infringe any third party's copyright, patent or other intellectual property rights. Except as warranted, the Software and Documentation is being provided "as is." THE FOREGOING LIMITED WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES OR CONDITIONS, EXPRESS OR IMPLIED, AND THE FOUNDRY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES OR CONDITIONS, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, REGARDLESS OF WHETHER THE FOUNDRY KNOWS OR HAS REASON TO KNOW OF LICENSEE'S PARTICULAR NEEDS. The Foundry does not warrant that the Software or Documentation will meet Licensee's requirements or that Licensee's use of the Software will be uninterrupted or error free. No employee or agent of The Foundry is authorized to modify this limited warranty, nor to make additional warranties. No action for any breach of the above limited warranty may be commenced more than one (1) year after Licensee's initial receipt of the Software. To the extent any implied warranties may not be disclaimed under applicable law, then ANY IMPLIED WARRANTIES ARE LIMITED IN DURATION TO NINETY (90) DAYS AFTER DELIVERY OF THE SOFTWARE TO LICENSEE.

SECTION 9. LIMITED REMEDY.

The exclusive remedy available to the Licensee in the event of a breach of the foregoing limited warranty, TO THE EXCLUSION OF ALL OTHER REMEDIES, is for Licensee to destroy all copies of the Software, send The Foundry a written certification of such destruction and, upon The Foundry's receipt of such certification, The Foundry will make a replacement copy of the Software available to Licensee.

SECTION 10. INDEMNIFICATION.

Licensee agrees to indemnify, hold harmless and defend The Foundry and The Foundry's affiliates, officers, directors, shareholders, employees, authorized resellers, agents and other representatives (collectively, the "Released Parties") from all claims, defense costs (including, but not limited to, attorneys' fees), judgments, settlements and other expenses arising from or connected with the operation of Licensee's business or Licensee's possession or use of the Software or Documentation.

SECTION 11. LIMITED LIABILITY.

In no event shall the Released Parties' cumulative liability to Licensee or any other party for any loss or damages resulting from any claims, demands or actions arising out of or relating to this Agreement (or the Software or Documentation contemplated herein) exceed the License Fee paid to The Foundry or its authorized reseller for use of the Software. Furthermore, IN NO EVENT SHALL THE RELEASED PARTIES BE LIABLE TO LICENSEE UNDER ANY THEORY FOR ANY INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, EXEMPLARY OR CONSEQUENTIAL DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS OR LOSS OF PROFITS) OR THE COST OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, REGARDLESS OF WHETHER THE RELEASED PARTIES KNOW OR HAVE REASON TO KNOW OF THE POSSIBILITY OF SUCH DAMAGES AND REGARDLESS OF WHETHER ANY REMEDY SET FORTH HEREIN FAILS OF ITS ESSENTIAL PURPOSE. No action arising out of or related to this Agreement, regardless of form, may be brought by Licensee more than one (1) year after Licensee's initial receipt of the Software; provided, however, to the extent such one (1) year limit may not be valid under applicable law, then such period shall limited to the shortest period allowed by law.

SECTION 12. TERM; TERMINATION.

This Agreement is effective upon Licensee's acceptance of the terms hereof (by clicking on the "Accept" button on installation) and Licensee's payment of the License Fee, and the Agreement will remain in effect until termination. If Licensee breaches this Agreement, The Foundry may terminate the License granted hereunder by notice to Licensee. In the event the License is terminated, Licensee will either return to The Foundry all copies of the Software and Documentation in Licensee's possession or, if The Foundry directs in writing, destroy all such copies. In the later case, if requested by The Foundry, Licensee shall provide The Foundry with a certificate signed by an officer of Licensee confirming that the foregoing destruction has been completed.

SECTION 13. CONFIDENTIALITY.

Licensee agrees that the Software and Documentation are proprietary and confidential information of The Foundry and that all such information and any communications relating thereto (collectively, "Confidential Information") are confidential and a fundamental and important trade secret of The Foundry. Licensee shall disclose Confidential Information only to Licensee's employees who are working on an Authorized Project and have a "need-to-know" such Confidential Information, and shall advise any recipients of Confidential Information that it is to be used only as authorized in this Agreement. Licensee shall not disclose Confidential Information or otherwise make any Confidential Information available to any other of Licensee's employees or to any third parties without the express written consent of The Foundry. Licensee agrees to segregate, to the extent it can be reasonably done, the Confidential Information from the confidential information and materials of others in order to prevent commingling. Licensee shall take reasonable security measures, which such measures shall be at least as great as the measures Licensee uses to keep Licensee's own confidential information secure (but in any case using no less than a reasonable degree of care), to hold the Software, Documentation and any other Confidential Information in strict confidence and safe custody. The Foundry may request, in which case Licensee agrees to comply with, certain reasonable security measures as part of the use of the Software and Documentation. Licensee acknowledges that monetary damages may not be a sufficient remedy for unauthorized disclosure of Confidential Information, and that The Foundry shall be entitled, without waiving any other rights or remedies, to such injunctive or equitable relief as may be deemed proper by a court of competent jurisdiction.

SECTION 14. INSPECTION.

Licensee shall advise The Foundry on demand of all locations where the Software or Documentation is used or stored. Licensee shall permit The Foundry or its authorized agents to inspect all such locations during normal business hours and on reasonable advance notice.

SECTION 15. NONSOLICITATION.

Licensee agrees not to solicit for employment or retention, and not to employ or retain, any of The Foundry's current or future employees who were or are involved in the development and/or creation of the Software.

SECTION 16. U.S. GOVERNMENT LICENSE RIGHTS.

The Software, Documentation and/or data delivered hereunder are subject to the terms of this Agreement and in no event shall the U.S. Government acquire greater than RESTRICTED/LIMITED RIGHTS. At a minimum, use, duplication or disclosure by the U.S. Government is subject to the applicable restrictions of: (i) FAR §52.227-14 ALTS I, II and III (June 1987); (ii) FAR §52.227-19 (June 1987); (iii) FAR §12.211 and 12.212; and/or (iv) DFARS §227.7202-1(a) and DFARS §227.7202-3.

The Software is the subject of the following notices:

- Copyright (c) 2010 The Foundry Visionmongers Ltd. All Rights Reserved.
- Unpublished-rights reserved under the Copyright Laws of the United Kingdom.

SECTION 17. SURVIVAL.

Sections 2, 4, 5, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 18 and 19 shall survive any termination or expiration of this Agreement.

SECTION 18. IMPORT/EXPORT CONTROLS.

To the extent that any Software made available hereunder is subject to restrictions upon export and/or reexport from the United States, Licensee agrees to comply with, and not act or fail to act in any way that would violate, the applicable international, national, state, regional and local laws and regulations, including, without limitation, the United States Foreign Corrupt Practices Act, the Export Administration Act and the Export Administration Regulations, as amended or otherwise modified from time to time, and neither The Foundry nor Licensee shall be required under this Agreement to act or fail to act in any way which it believes in good faith will violate any such laws or regulations.

SECTION 19. MISCELLANEOUS.

This Agreement is the exclusive agreement between the parties concerning the subject matter hereof and supersedes any and all prior oral or written agreements, negotiations, or other dealings between the parties concerning such subject. This Agreement may be modified only by a written instrument signed by both parties. If any action is brought by either party to this Agreement against the other party regarding the subject matter hereof, the prevailing party shall be entitled to recover, in addition to any other relief granted, reasonable attorneys' fees and expenses of litigation. Should any term of this Agreement be declared void or unenforceable by any court of competent jurisdiction, such declaration shall have no effect on the remaining terms of this Agreement. The failure of either party to enforce any rights granted hereunder or to take action against the other party in the event of any breach hereunder shall not be deemed a waiver by that party as to subsequent enforcement of rights or subsequent actions in the event of future breaches. This Agreement shall be governed by, and construed in accordance with English law.

Copyright (c) 2010 The Foundry Visionmongers Ltd. All Rights Reserved. Do not duplicate.

INDEX

A-Z

Numerics
2.5D transformations 236
2D transformations 226
3D 788
3D geometry 806
3D objects
  normals 476
3D scenes, lighting 470
3D tracking 661
3D viewers 788
3D workspace 426

A
Add nodes 221
adding
  a black edge to the bounding box 170
  Axis objects to 3D scenes 441
  Card objects to 3D scenes 433
  Cube objects to 3D scenes 436
  mathematical functions to expressions 537
  motion blur 240
  nodes 46
  OBJ-type objects to 3D scenes 438
  Sphere objects to 3D scenes 437
  viewers 88
AddTimeCode nodes 157
AdjBBox nodes 168
adjusting
  contrast 205
  gain 205
  gamma 205
  HSV 211
  hue 209
  offset 205
  saturation 209
  value 209
Alpha Bias 308
anaglyph images, converting into 508
Anaglyph nodes 508
analysing frame sequences 38, 422
analysing pixels 422
animating parameters 71
AppendClip nodes 384
applying
  image formats 164
  motion blur 382
  tracking data to clips 256
AVI format 829
Axis nodes 442
  adding to scenes 441

B
Backdrop nodes 142
BasicMaterial nodes 812
Bezier nodes 329
Bezier shape 343
Bezier Tool 344
Biasing 306
Bicubic nodes 433
black pixels
  See Status 301
black points, sampling 204
BlackOutside nodes 170
blending mode 354
blending modes 342, 343
blue pixels
  See Status 304
Blur Tool 339
blurring 339
bounding box
  adding a black edge to 170
  copying from an image 169
  resizing 168
Brush Tool 335
B-Spline shape 343
B-Spline Tool 346
Burn Tool 343

C
calling, channels 174
cameras
  dollying 430
  editing lens characteristics of 478
  resetting 431
  rolling 430
  tracking 661
CameraTracker nodes 661
Card nodes 433
  adding to 3D scenes 432
CCorrect nodes 206
changing
  convergence 510
channel count 142
channel values
  applying expressions to 222
  clamping 220
  inverting 221
  multiplying 222
  offsetting 221
channels 172
  applying mathematical operations to 220
  calling 174
  creating 173
  deleting 179
  displaying in a viewer 94
  renaming 178
  separating 65
  swapping 180
  tracing 178
CIN format 829
Cineon conversions 224
Clamp nodes 220
clamping channel values 220
Clean BG Noise 264
Clean FG noise 265
Clip Black 309
Clip Rollback 313
Clip White 309
clips
  applying tracking data to 256
  corner-pinning 257
  cutting frames from 383
  dissolving between 384
  editing 382
  fading to black 384
  retiming 371
  slipping 382
  splicing 384
  stabilizing 259
  warping 379

Clone Tool 336
cloning nodes 51
color curves 206
color picker 67
color pickers 65
Color Replacement 317
color sliders 67
color space conversions 223
color wheel 67
Colorspace nodes 224
CompareMetaData nodes 154
configuring Nuke 603
connecting
  nodes 52
  viewers 88
contrast, adjusting 205
control panels 66
  displaying 63
convergence
  changing 510
CopyBBox nodes 169
copying
  a bounding box from an image 169
  a rectangle from an image 198
  nodes 50
CopyMetaData nodes 157
CopyRectangle nodes 198
CornerPin2D nodes 257
  applying tracking data to 259
corner-pinning clips 257
creating
  bg reflections on fg elements 406
  channels 173
  effects 406
  formats with the Reformat node 162
  layers 173
  star filter effects 409
Crop nodes 167
cropping elements 166
CrosstalkGeo nodes 463
Cube nodes 437
  adding to 3D scenes 436
Cubic filtering algorithm 228
curve editor 73
curve list 331
curves, editing tracks with 252
CurveTool nodes 38, 422
curving points, smoothing points 365
cusping points 365
custom plug-ins 629, 643
customer support 23
customising
  Nuke 603
cutting
  a clip's frames 383
  nodes 50

D

defining
  common image formats 629
  common menu options 622
  common preferences 644
  tonal range 203
Degrain tools, Primatte 278
delete preferences 646
deleting
  channels 179
  image formats 164
  keyframes 87
  layers 179
  nodes 52
  shape/stroke 364
  shape/stroke point 364
depth 692
  calculating 692
depth map 692
DepthGenerator nodes 692
Despot 314
Dilate 314
DirectLight nodes 471
disabling nodes 52
DisplaceGeo nodes 466
display gain 102
display gamma 102
displaying node parameters 63
displaying script information 142
dissolving between clips 384
distorting time 371
Dither nodes 216
Dodge Tool 342
Dot nodes 54
DPX format 829
dual monitors 116
duplicate
  curves 352
  stroke/shape 332
DV footage 312

E

editing
  camera lens characteristics 478
  clips 382
  image formats 164
  nodes 50
  parameters 63
  tracks with curves 252
  tracks with overlays 252
elements
  cropping 166
  rotating in 2.5D 238
  rotating in 2D 232
  scaling in 2.5D 239
  scaling in 2D 233
  skewing in 2.5D 239
  skewing in 2D 235
  translating in 2.5D 238
  translating in 2D 231
environment light 473
Environment nodes 473
Eraser Tool 335
expressions 533
  adding mathematical functions to 537
  applying to channel values 222
  applying tracking data with 256
  linking 533
  syntax of 533
Expressions nodes 222
EXPTool nodes 740
EXR format 829

F

feather
  adding 358
  removing 358
FFT 26
file formats, supported 829
file name conventions 528
FilmGrain nodes 216
filtering algorithms 237
  selecting 227
fine tuning sliders, Primatte 284
flipbooking sequences 518, 741
formats
  AVI 829
  CIN 829
  creating with the reformat node 162
  DPX 829
  EXR 829
  FPI 829
  GIF 829
  HDRI 829

- IFF 830
- JPEG 830
- MOV 830
- PIC 830
- PNG 830
- PSD 830
- RAW 830
- REDCODE 830
- RLA 831
- SGI 830
- Targa 831
- TIFF 831
- XPM 831
- YUV 831

Foundry Flexlm Tools 26
four-point tracking 747
FPI format 829
Frame Blend nodes 373
frame ranges 140
FrameCycler 519
FrameRange nodes 383
frames
  cutting from clips 383
  interpolating between for temporal effects 372
full size formats 122
full size mode 122
full-size formats 122
full-size mode 124

G

gain
  adjusting 205
  displayed in a viewer 102
gamma
  adjusting 205
  displayed in a viewer 102
garbage mask 315
generating, tracks 250
GIF format 829
gizmos 629
Glint nodes 409
global frame range 381
Grade nodes 204
grain
  matching 216
  practical 217
  synthetic 216
Grain nodes 216
grainy footage 312
gray pixels
  See Status 301
green pixels
  See Status 303
GridWarp nodes 386, 398, 401

H

HDRI format 829
help, online 22
histograms 203
Host ID number 27
hot keys 816
HSV, adjusting 211
HSVTool nodes 211
hue, adjusting 209
HueCorrect nodes 213

I

IBK 768
IFF format 830
iitter, removing from tracks 253
image formats 162
  applying 164
  defining common 629
  deleting 164
  displaying in a viewer 97
  editing 164
  supported 829
Image-Based Keyer 768
images
  scaling up in proxy mode 127
Impulse filtering algorithm 228
indicators
  on nodes 55
input channels, selecting 178
input fields 64
input processes 103
Inside Mask 315
Inside Masks 315
interface preferences 543
Invert nodes 221
inverting channel values 221

J

JoinViews nodes 503
JPEG format 830

K

keyboard shortcuts 816
Keyer nodes 778
keyframes
  deleting 87
  setting 72
keying 261, 746, 761
keying video 778
Keys filtering algorithm 229

L

launching Nuke 33, 713
layering images 183
layers 172
  creating 173
  deleting 179
lens
  distort 680
  type 684
  undistort 680
LensDistortion nodes 680
license keys 27
licenses
  floating 26
  node-locked 26
  obtaining 26
licensing 33
Light nodes 475
lighting 811
LightWrap nodes 406
linking expressions 533
lin-to-log conversions 223
lmhostid number 27
Log2Lin nodes 224
LogGeo nodes 465
Lookup nodes 208
lookup tables
  altering for a script 647
  default settings 650
LookupGeo nodes 463
LUTs
  altering for a script 647
  default settings 650

M

macros. See gizmos.
masking color corrections 214
masking effects 740
Masks 315
masks, selecting 176
matchmoving 746, 757
mathematical operations 220
Matte
  Rollback 313
Mattes 315
menu options, defining 622
Merge nodes 183
MergeGeo nodes 443
merging images 183

MetaData 152
metadata
  working with 152
Mitchell filtering algorithm 230
MixViews nodes 500
ModifyMetaData nodes 154
monitors, dual 116
morphing 386, 400
motion blur, adding 240
motion blur, applying 382
MotionBlur3D nodes 482
MOV format 830
Multiply nodes 222
multiplying channel values 222

N
naming, rendered images 528
node count 142
nodes
  adding 46
  cloning 51
  connecting 52
  copying 50
  cutting 50
  decloning 52
  deleting 52
  disabling 52
  editing 50
  pasting 51
  previewing output from 517
  reconnecting 53
  replacing 49
  selecting 47
Notch filtering algorithm 230
nudge
  nudging a point 364
  nudging clone source 337

O
objects
  Axis type 441
  Card type 432
  controlling display of in viewer 456
  Cube type 436
  OBJ type 438
  skewing 459
  Sphere type 437
offset, adjusting 205
offsetting
  channel values 221
  tracks 251
OFlow nodes 376
one-point tracking 747
OneView nodes 503
online help 22
Opacity 354
Outside Component 315
Outside Mask 315
Outside Masks 315
overlays, for tracks 252

P
Paint nodes 329
painting 335
panning
  in the node graph 58
  in viewers 89
parameters
  animating 71
  displaying for nodes 63
  editing 63
Parzen filtering algorithm 230
pasting, nodes 51
pausing, viewers 94
Phong nodes 446
Photoshop format 830
PIC format 830
picking the screen color 304
pivot points 460
pixel analyzer 422
pixel aspect ratio 100, 162
pixel values 97
playback speed of clips 381
PNG format 830
point light 471
Point nodes 471
Position nodes 232
postage stamps, generating 162
practical grain 217
PreBlur 312
preface 22, 658
preferences 543
  defining common 644
  reset 646
preview
  flipbook 517
  region of interest (ROI) 517
previewing outputs 517
Primatte 261
  algorithm 273, 287
  Auto-Compute 262
  Degrain tools 278
  RT 296
  RT Algorithm 274
  RT+ 295
  RT+ Algorithm 273
  Select BG Color 263
Primatte nodes 763
ProcGeo nodes 467
Project3D nodes 448
proxy files 126
proxy formats 124
proxy mode 98, 124, 736
  scaling up images 127
proxy scale 124
PSD format 830
PyQt 600
Python 561
Python statements 566

Q
QuickTime format 830

R
Radial Distortion 684
RadialDistort nodes 468
Ramp nodes 722
RAW format 830
ReadGeo nodes 439
reconnecting
  nodes 53
red pixels
  See Status 304
REDCODE format 830
redoing actions 87
Reformat nodes 162, 735
reformatting elements 162
region of interest 101, 517
removing
  jitter from tracks 253
  spill 266, 267, 268
renaming channels 178
renaming nodes 49
render
  farms 532
  format 522
  resolution 522
rendering 742
  3D scenes 428
  a sequence to flipbook 518
  from single Write nodes 524
  scripts 517, 522
repeatable sampling, Primatte 268
replacing nodes 49
resizing the bounding box 168

resolution 162
  define custom 523
Retime nodes 371
retiming
  clips 371
  with OFlow 376
Reveal Tool 337
Rifmen filtering algorithm 229
rippling keyframes 368
RLA format 831
ROI (region of interest) 101, 517
Rollback 313
rolling, cameras 430
rotating
  cameras 430
  elements in 2.5D 238
  elements in 2D 232
RotoPaint node 773
RotoPaint nodes 329
RotoPaint tools
  default 348
rotoscoping 773

S

sampling
  black points 204
  white points 204
Saturation nodes 214
saturation, adjusting 209
scaling elements
  in 2.5D 239
  in 2D 233
ScanlineRender nodes 428, 789
Scene nodes 427, 797
scenes
  adding Axis objects to 441
  adding Cube objects to 436
  adding OBJ-type objects to 438
  adding Sphere objects to 437
Screen Balance 311
Screen Color 304
Screen Despot Black 314
Screen Despot White 314
Screen Dilate 314
Screen Gain 301, 310
Screen Replace 317
Screen Replace Color 317
Screen Strength 310
script editor 562
script information
  displaying 142
scripting languages, Python 561
scripts 110
  global frame range 381
  playback speed 381
  rendering 517, 522
searching
  for nodes 56
Seeding Tracks 662
selecting
  filtering algorithms 227
  masks 176
  nodes 47
setting keyframes 72
setting up
  scripts 122
  tracks 246
  views 494
SGI format 830
sharpen 340
Sharpen Tool 340
ShuffleViews nodes 507
Simon filtering algorithm 229
skewing
  elements in 2.5D 239
  elements in 2D 235
  objects 459
sliders 65
slipping clips 382
Smear Tool 341
smoothing tracks 253
Source Alpha 316
spacebar 118
Sphere nodes 437
Sphere objects
  adjusting geometric resolution of 438
spill
  removing 266, 267, 268
  replacement methods 270, 286
  suppressing 213
splicing clips 384
SplineWarp nodes 393, 399, 403
SplitAndJoin 503
Spotlight nodes 472, 813
stabilising 746, 754
Stabilize2D nodes
  applying tracking data to 260
  stabilizing clips with 260
stabilizing clips 259
Status 302
stereoscopic images
  previewing 515
  reading 496
  rendering 515
stereoscopic projects 494
  displaying views 498
  editing views 500
  setting up views 494
StickyNote nodes 146
stroke/shape list 331
supported file formats 829
swapping channels 180
synchronising viewer playback 93
synthetic grain 216
system requirements 25

T

Targa format 831
TCL 533, 629, 644
  using in expressions 537
temporal operations 371
Text nodes 412
third party licenses 839
three-point tracking 747
TiCkLe 533, 629, 644
TIFF format 831
Time Cut nodes 383
time distortion 371
time-based operations 371
TimeBlend nodes 374
TimeBlur nodes 382
timeline controls 91
TimeOffset nodes 383
tonal range
  defining 203
  defining with a histogram 204
tracing channels 178
Tracker nodes 246, 248, 746
tracking 746
  per vertex 256
tracking specific points 662
tracks
  editing 252
  generating 250
  offsetting 251
  pattern area for 249
  positioning anchors for 249
  search area for 249
  smoothing 253
Transform nodes 231, 233, 234, 235
Transform2D nodes 233
Transform3D nodes 236, 238
transformation handles 237
transformations 226

  - choosing order of operations for 459
  - in 2.5D 236
  - in 2D 226
  - moving pivot for objects 460
  - skewing objects 459
- TransformGeo nodes 457
- translating
  - a Cube object's sides 437
  - cameras 430
  - elements in 2.5D 238
  - elements in 2D 231
- Trilinear nodes 469
- two-point tracking 747

U

- undistort input 668, 704
- undoing actions 87
- UVProject nodes 447

V

- value, adjusting 209
- vertex tracking 256
- View 302
- viewer input processes 651
- viewer playback, synchronising 93
- viewer processes 103
- viewers 87, 731
  - adding 88
  - changing colorspace 225
  - comparison wipes 108
  - configuring display of 3D objects in 456
  - connecting 88
  - display composite models 108
  - displaying views 498
  - panning 89
  - pausing 94
  - using the controls 90
  - zooming 89
- ViewMetaData nodes 153
- views, setting up for project 494

W

- warping 386
- warping clips 379
- white pixels
  - See Status 301
- white points, sampling 204
- window layouts 116, 716
- wipe control in viewers 108
- Write nodes 523
- write on 361
- wxPython 597

X

- XPM format 831

Y

- YUV format 831

Z

- z-depth 692
  - calculating 692
- zooming
  - in the node graph 59
  - in viewers 89